Distributed Systems

2. Networking

Werner Nutt

2. Networking

2.1 Network Types and Performance

Performance Measures

- Latency (unit s)
  - determined by software overheads, current load
- Bandwidth (byte/s)
  - determined by characteristics of network

  (in DS, latency usually more important, since messages are small)
- Total system bandwidth
  - in LANs, usually equal to bandwidth
- Bandwidth utilization
- Round Trip Time (RTT)
  - time needed to send a message and receive a reply

  How can one measure that?

Network Performance: Examples

- Method invocation within a process: 1 μs
- RTT for request/reply in LAN: 1 ms
- Access to file in cached memory on file server: 10 ms
- Access to file on local disk: 10 ms
- RTT on Internet: 50-500 ms

Types of Network

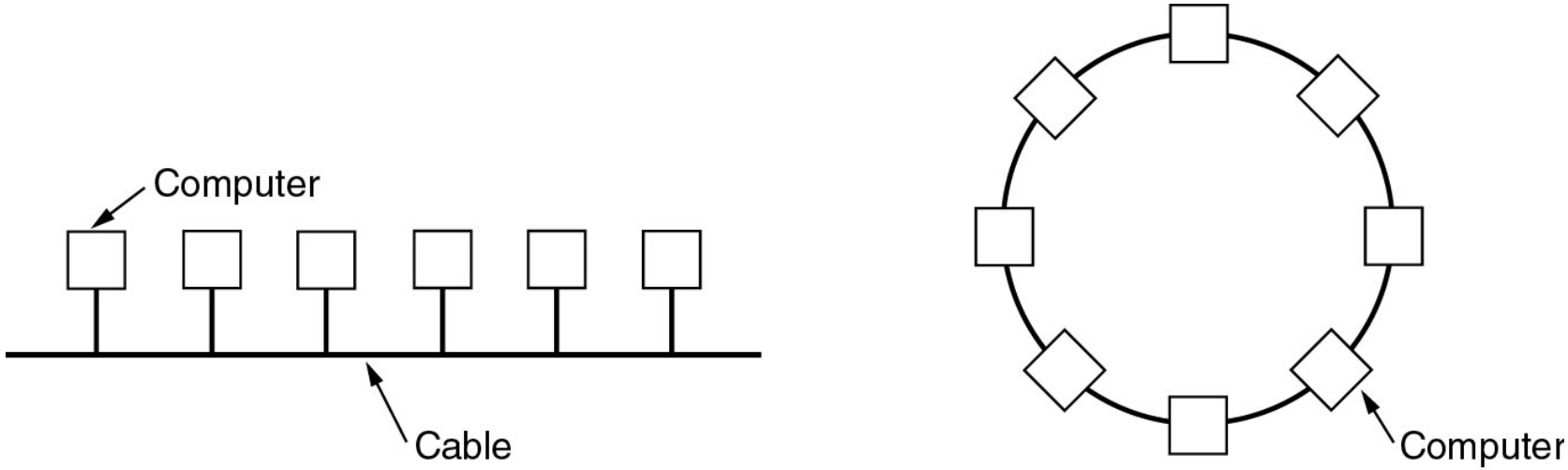
- LANs (Local Area Networks)
  - technology suitable for small area (building, campus)
  - usually wire
- MANs (Metropolitan Area Networks)
  - intra-city, cable based, multimedia
- WANs (Wide Area Networks)
  - large distances, inter-city/country/continental
- Wireless networks
  - WLANs, WPANs (= wireless personal area networks)

Distinguished by technology, not only distances

LANs

- High bandwidth
- Low latency
- Technology
  - predominantly Ethernet, now 100/1000Mbps
  - earlier token ring
  - also ATM, better QoS (= Quality of Service), but more expensive
- Topology
  - Network may consist of subnetworks
  - Segments connected by hubs (1 in, all out) and switches (1 in, 1 out)

LAN Topology



(a) (b)

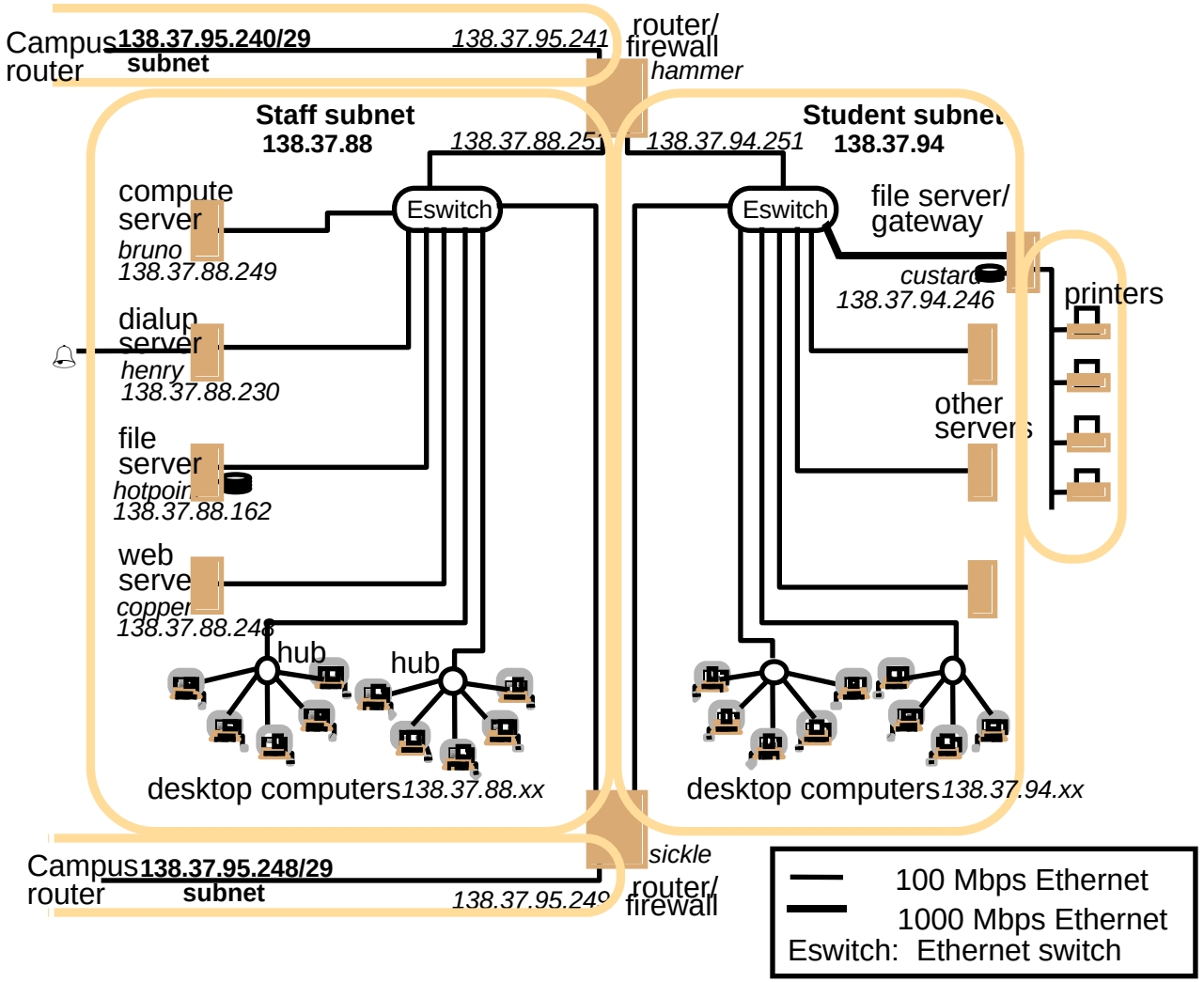
Broadcast Networks:

(a) Bus

(b) Ring

Example of a University LAN

Simplified view of the Queen Mary and Westfield College Computer Science network

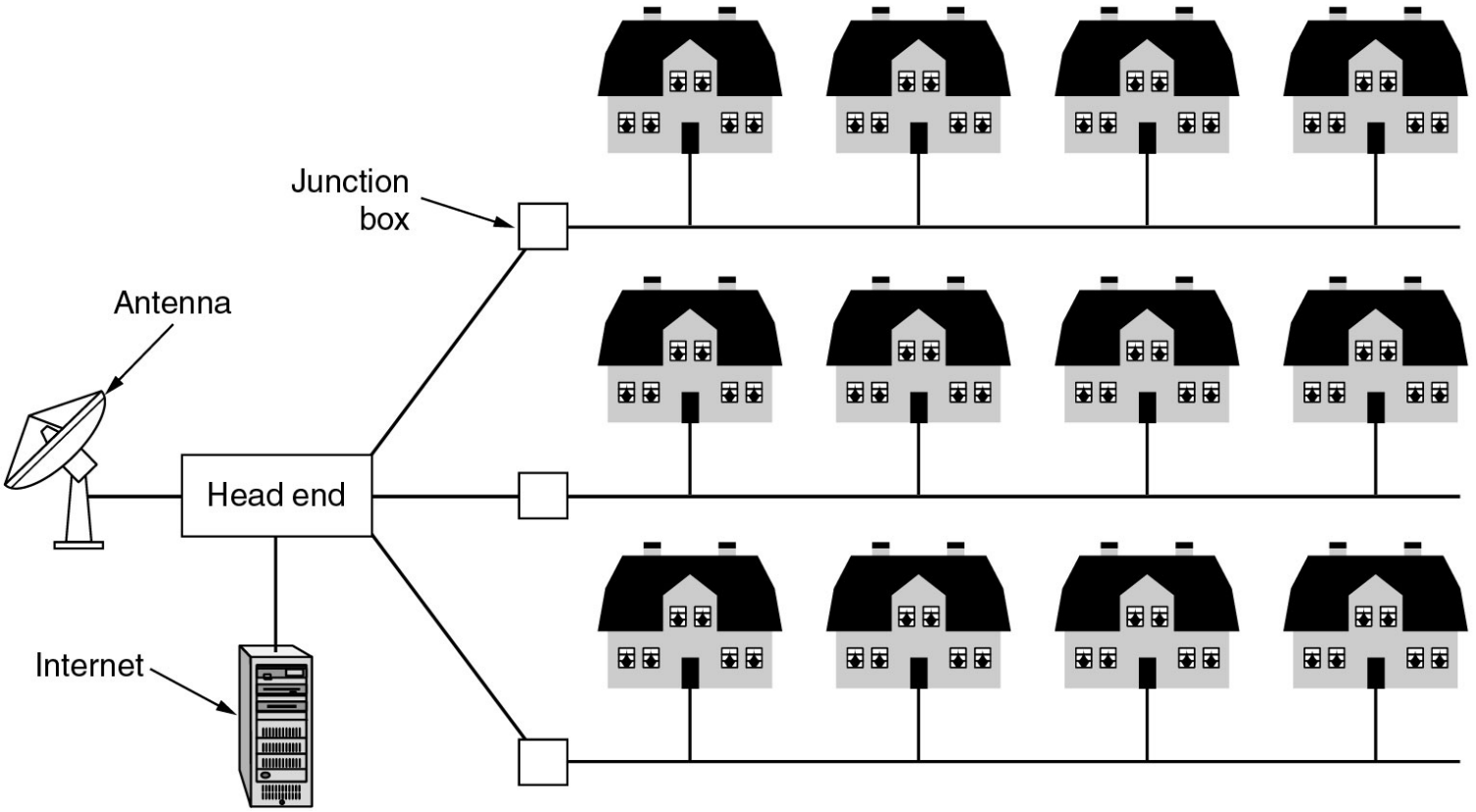


(Ethernet at FUB is switched)

MANs

- Connect several LANs
  - also “Campus Networks”
- Wire/fibre cable networks
- Transmission based on
  - Ethernet
  - ATM (= Asynchronous transfer mode)
- Examples:
  - Cable TV networks

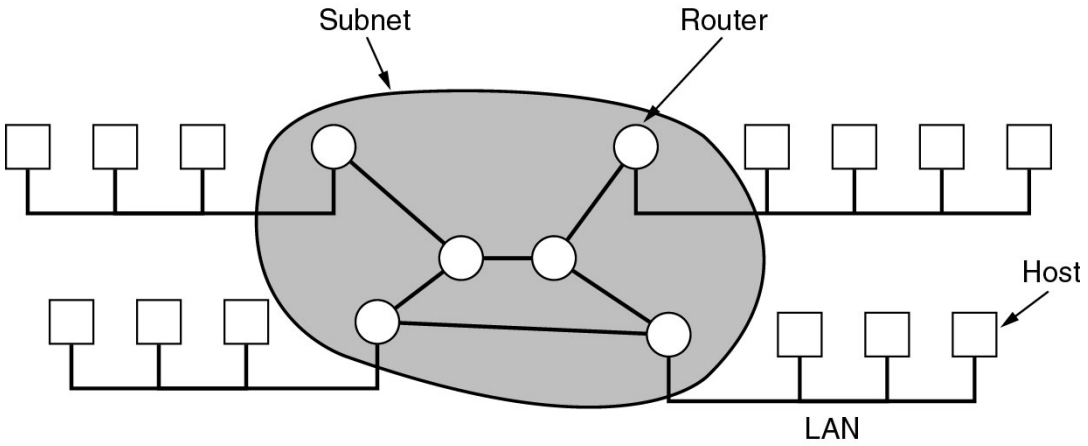
MAN Based on Cable TV



Bandwidth not used for TV
is made available for data transmission

WANs

- Links smaller networks (LANs, MANs) using a communication subnet
  - high-speed transmission lines, possibly leased (satellite/wire/fibre)
  - routers



➔ fast medium, but
  - signals travel long distances
    - Europe-Australia 0.1 s
    - via geostationary satellite 0.2 s
  - routers introduce delays

Example: Networks between research and academic institutions

Wireless Networks

- WLANs (Wireless Local Area Networks)
  - extending and coexisting with wired LANs
  - WaveLAN technology (IEEE 802.11)

IEEE *("I triple E")* = Institute of Electric and Electronic Engineers

- WPANs (Wireless Personal Area Networks)
  - variety of technologies:
    - Infra-red,
    - BlueTooth low-power radio
- Mobile Phone Networks
  - GSM (= Global System for Mobile communication)

Network Performance

	Example	*Range*	*Bandwidth (Mbps)*	*Latency (ms)*
Wired:				
LAN	Ethernet	1-2 km	10-1000	1
MAN	ATM	250 km	1-150	10
WAN	IP routing	worldwide	0.010-600	50-500
Internetwork	Internet	worldwide	0.5-600	50-500
Wireless:				
WPAN	Bluetooth (802.15.1)	10 - 30m	0.5-2	5-20
WLAN	WiFi (IEEE 802.11)	0.15-1.5 km	2-54	5-20
WMAN	WiMAX (802.16)	550 km	1.5-20	5-20
WWAN	GSM, 3G phone nets	worldwide	0.01-02	100-500

2. Networking

2.2 Network Principles

Network Principles

- Transmission: Packet vs. Stream
- Switching schemes
- Protocol hierarchies

Transmission Modes

Packets

- Messages are divided into packets
  - of restricted length *(MTU = maximum transfer unit)*
  - with address of source and destination machine
- Packets are queued in buffers before sent onto link
  - packet loss if buffer overflow

➔ QoS not guaranteed

Data streaming

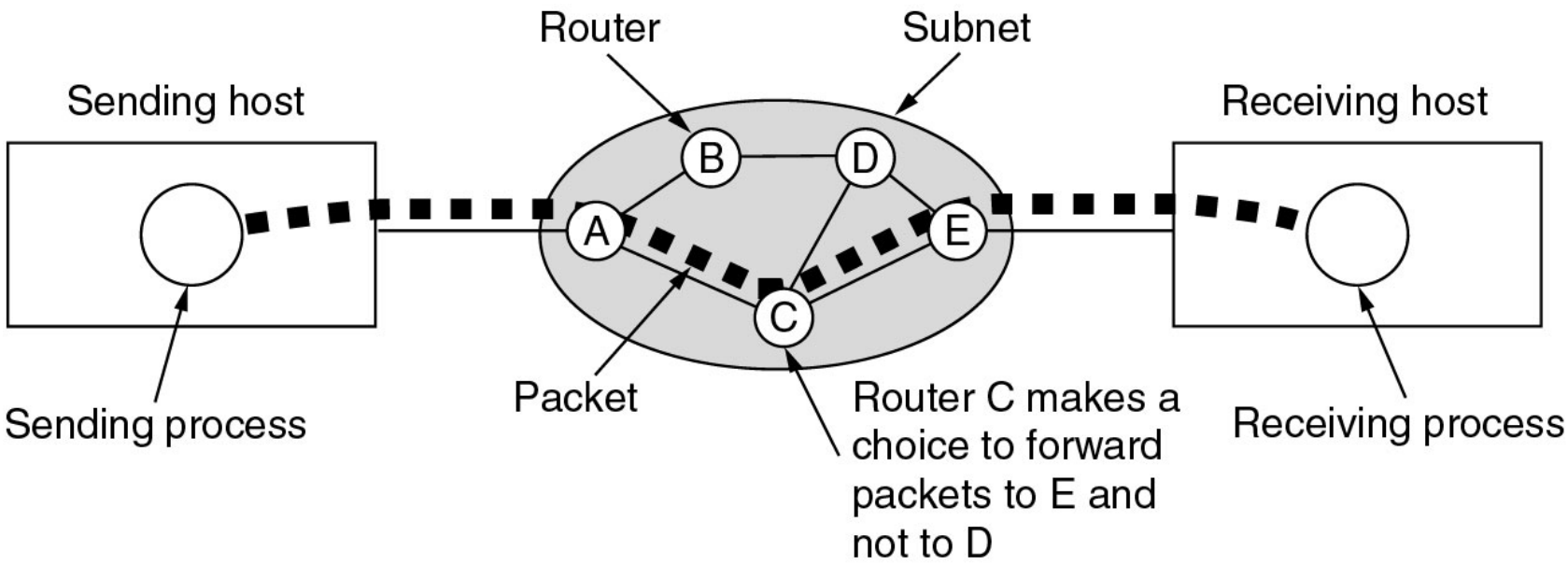
- Video, audio
- Needs high bandwidth, guaranteed maximal latency
- Often, a channel from sender to receiver is established
- Based on “packet” delivery with guaranteed QoS
- Example: ATM (Asynchronous Transfer Mode)

Switching Schemes (1)

Network = nodes connected by links

- Broadcasts *(Ethernet, wireless)*
  - send messages to all nodes
  - nodes listen for (other and own) messages ("carrier sensing")
- Circuit switching *(phone networks)*
  - establish path through network
  - physical change in the network connections
- Packet switching *(Internet Protocol)*
  - "store-and-forward"
  - unpredictable delays

Data Transport Based on Packet Routing

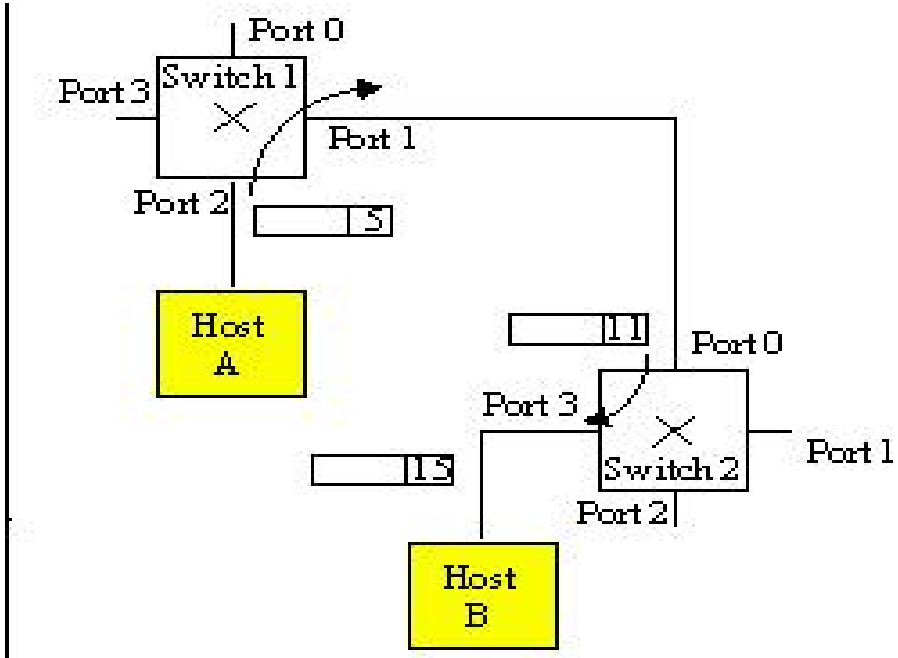


Switching Schemes (2)

- Virtual Circuit Switching (Frame/cell relay, e.g., *ATM)*
  - small, fixed size packets (48 byte of data for ATM),
  - padded if necessary
  - “logical” circuit switching
  - bandwidth & latency guaranteed (“virtual path”)
  - forwarding based on inspection of first few bytes
  - avoids error checking at nodes (uses reliable links)

- ATM
  - used by ISPs to realize (A)DSL

ATM: Virtual Path Set-Up



Switch	input port	incoming VCI	output port	outgoing VCI
1	2	5	1	11
2	0	11	3	15

Comparison: Circuit Switching, Packet Switching, and Virtual Circuit Switching

Circuit switching	Packet Switching	Virtual Circuit Switching
Dedicated transmission path	No dedicated path	No dedicated path
Continuous transmission of data	Transmission of packets	Transmission of packets
Messages are not stored	Packets may be stored until delivered	Packets stored until delivered
Path is established for entire conversation	Route established for each packet	Route established for entire conversation
Call setup delay, negligible transmission delay	Packet transmission delay	Call Setup delay, packet transmission delay

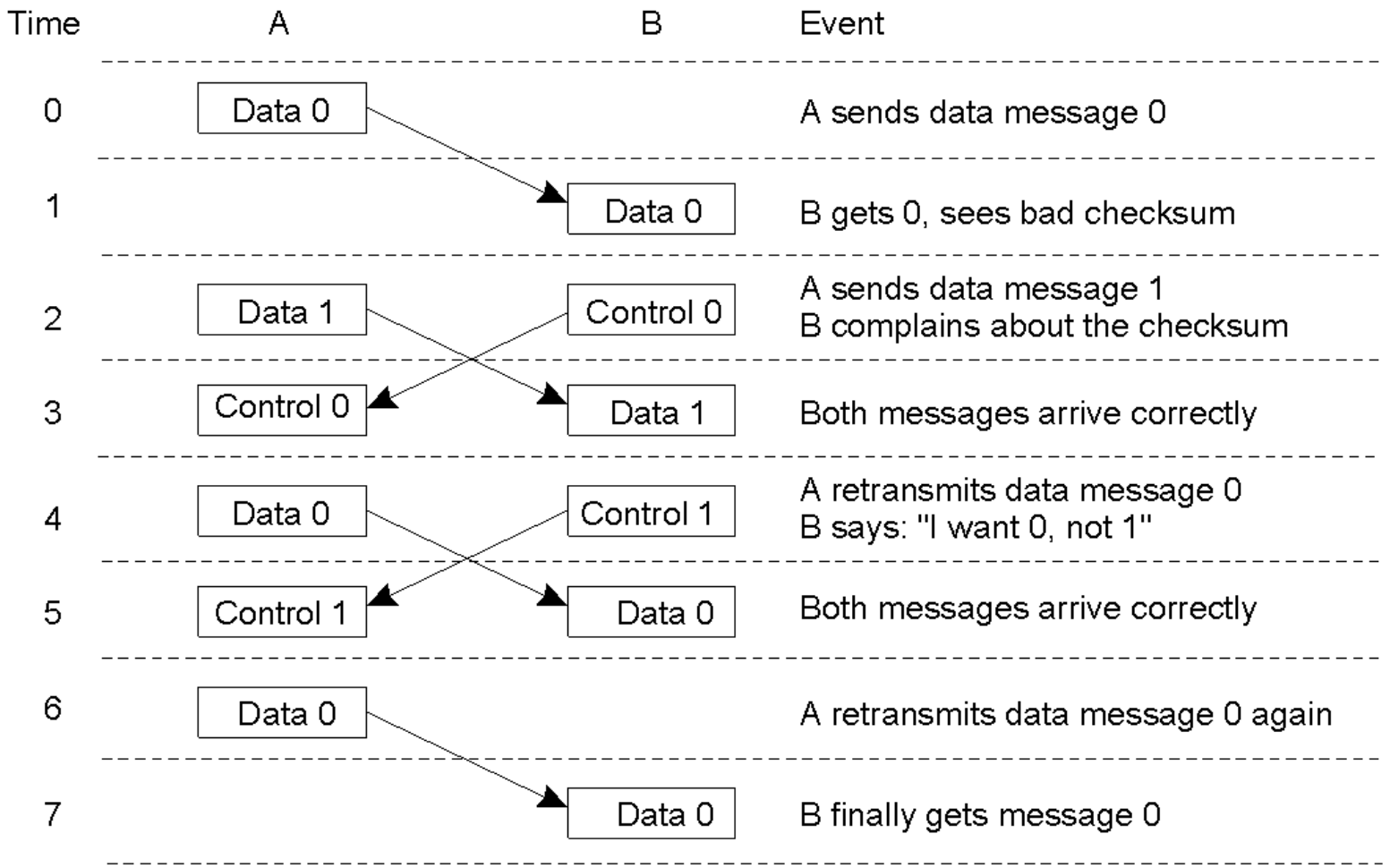
Comparison: Circuit Switching, Packet Switching, and Virtual Circuit Switching

Circuit switching	Packet Switching	Virtual Circuit Switching
Busy signal if called party busy	Sender may be notified if packet not delivered	Sender notified of connection denial
Overload may block call setup, no delay for established calls	Overload increases packet delay	Overload may block call setup, increases packet delay
Fixed bandwidth transmission	Dynamic use of bandwidth	Dynamic use of bandwidth
No overhead bits after call setup	Overhead bits in each packet	Overhead bits in each packet

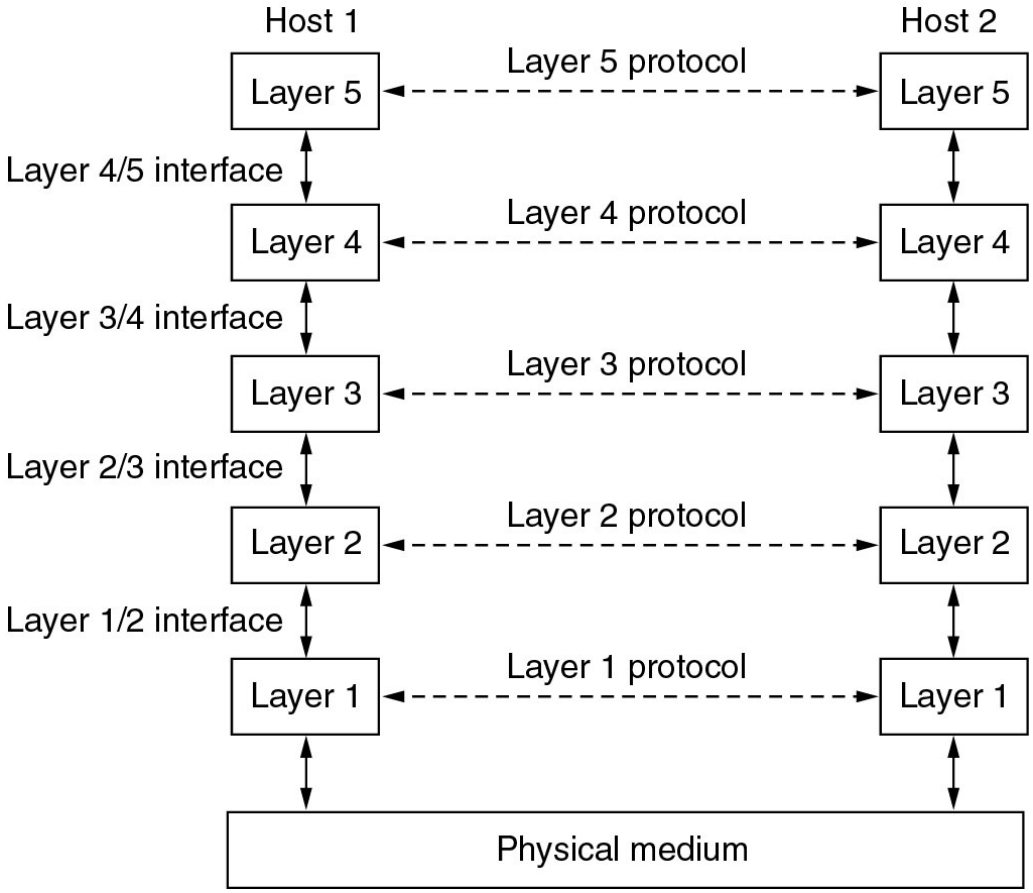
Protocols

- Set of conventions specifying
  - types and order of messages to be exchanged
  - data formats in messages
- Implemented by pairs of modules for
  - sending
  - receiving messages
- Arranged in layers
  - protocol suite/stack (= complete set of layers)

Fictitious Interaction Between Sender and Receiver



Protocols, Layers, Interfaces

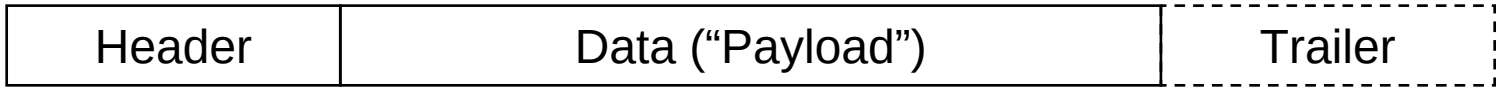


Protocol Stack

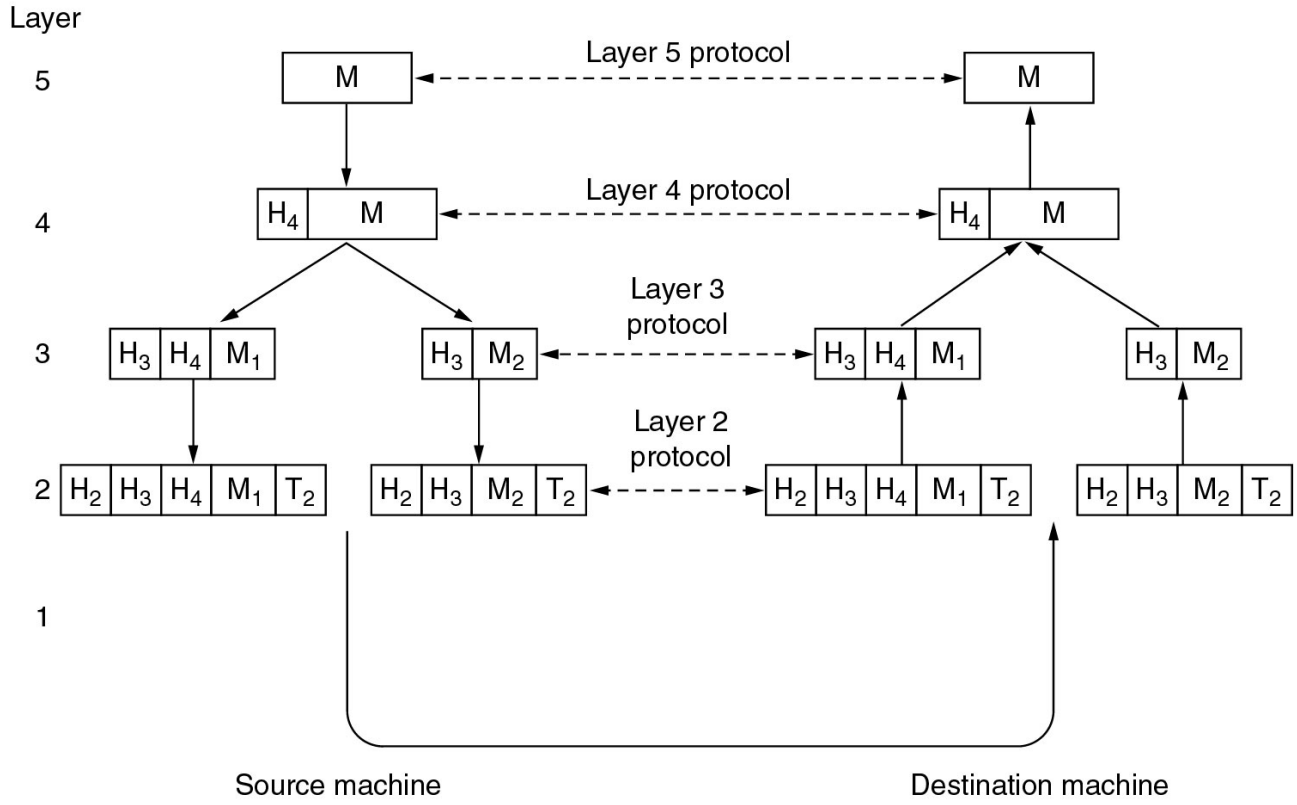
- Actual communication takes place only through the physical medium
- Each protocol layer offers virtual communication
- Interfaces define operations and services offered to higher layer

Information Flow Through Protocol Stack

- Each protocol specifies the structure of its packets



- Lower level packets carry higher level packets as their “payload”

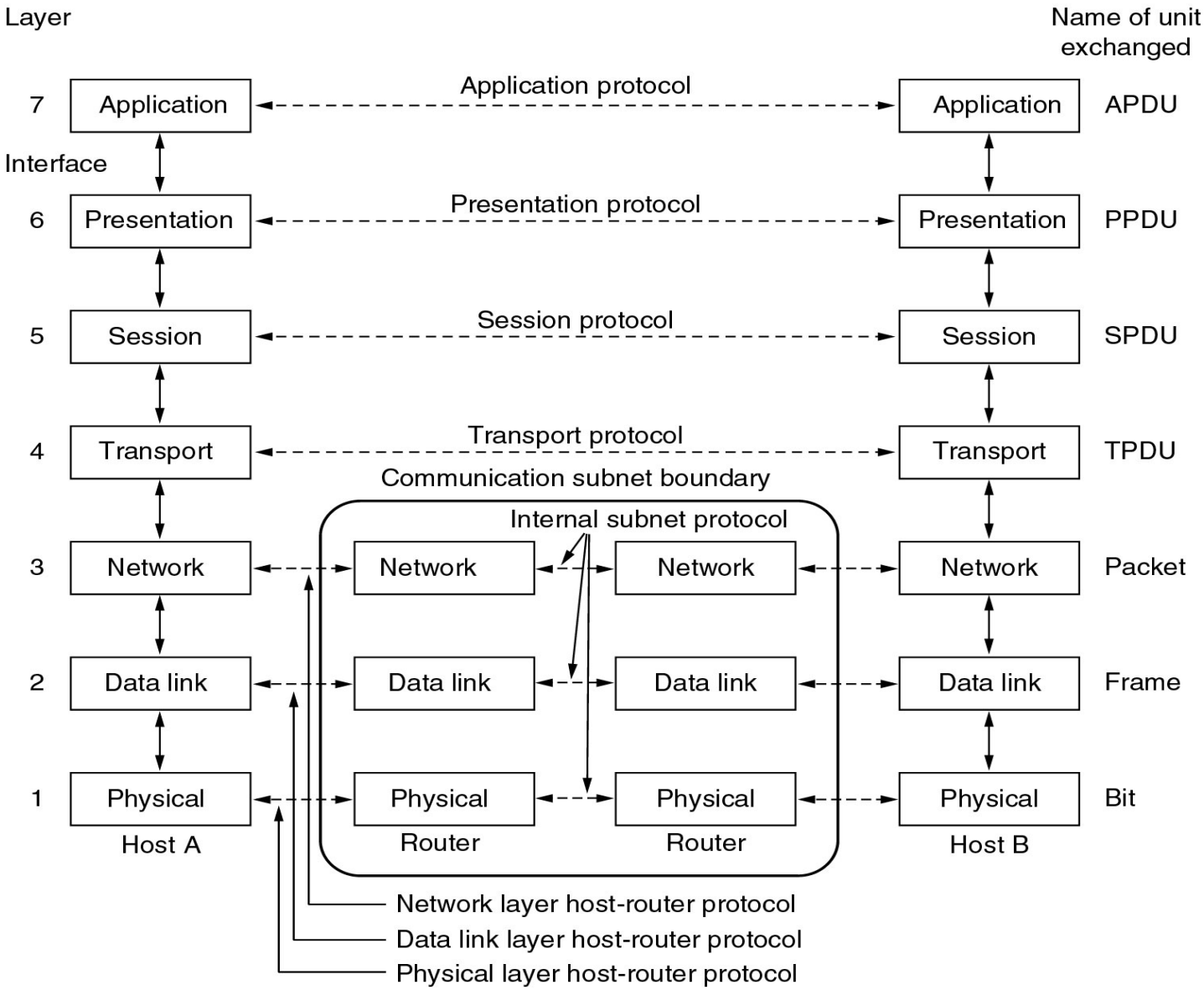


OSI Reference Model

OSI = Open Systems Interconnection

- Adopted by ISO (= International Standards Organisation) in 1977
- Framework for definition of protocols, not a standard
  - Since 1982, (failed) effort at defining compliant protocol suite
  - In parallel, definition of TCP/IP and Ethernet
- In reality, often one protocol encompasses several layers

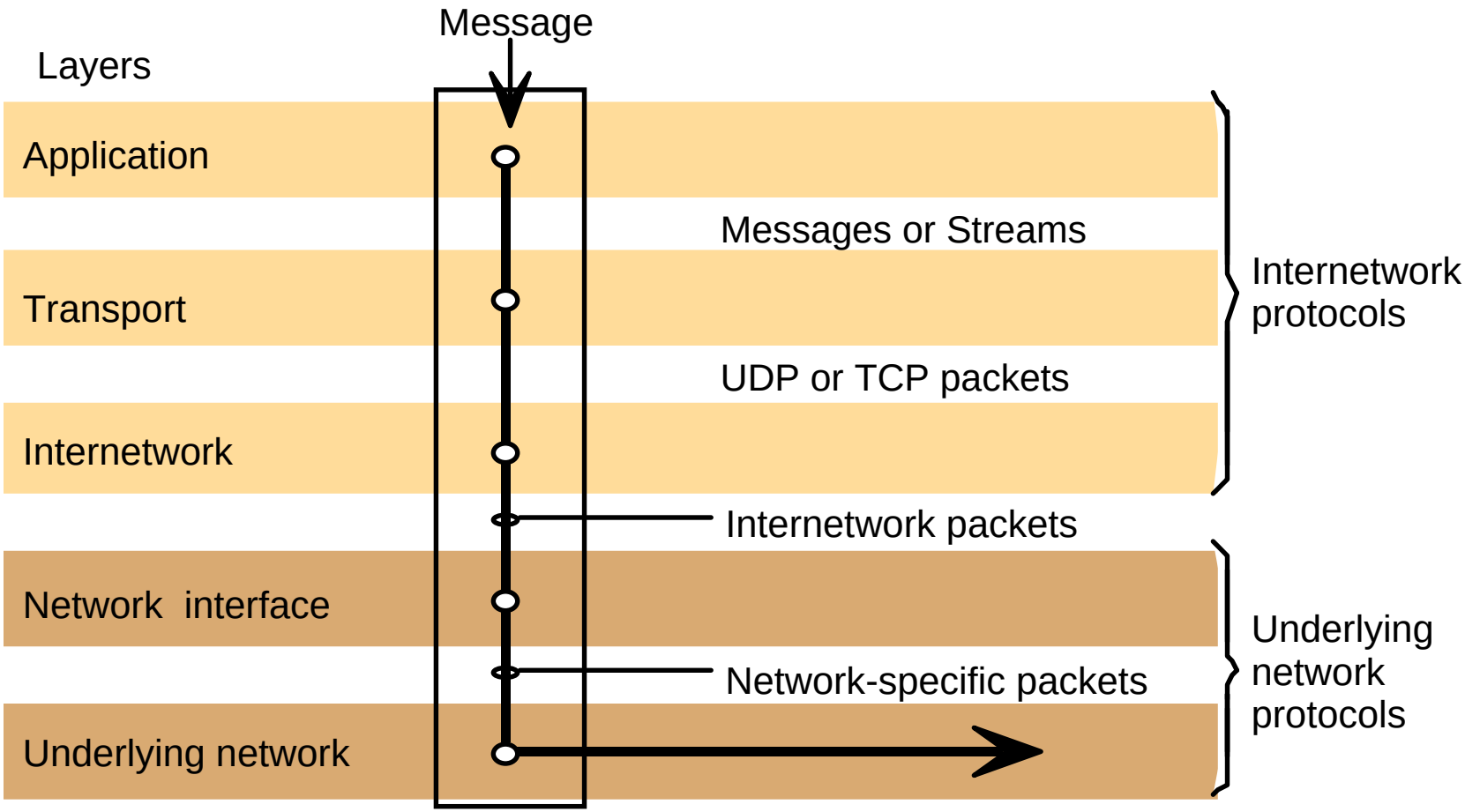
Layers in the OSI Model



OSI Protocol Suite (Summary)

Layer	*Description*	*Example*
Application	Protocols for specific applications.	HTTP, FTP, SMTP
Presentation	Protocols for independent data representation and encryption if required.	Secure Socket Layer (SSL), CORBA Data Rep.
Session	Protocols for failure detection and recovery.	
Transport	Message-level communication between ports attached to processes. Connection-oriented or connectionless.	TCP, UDP
Network	Packet-level transmission on a given network. Requires routing in WANs and Internet.	IP, ATM virtual circuits
Data link	Packet-level transmission between nodes connected by a physical link.	Ethernet Medium Access Control
Physical	Circuits and hardware driving the network	Ethernet signalling

TCP/IP Reference Model



2. Networking

2.3 Data Link Layer

Data Link Layer

Functions

- provides a service interface to the network layer
  - connectionless or connection-oriented
- deals with transmission errors
  - ➔ checksums
- regulates the data flow (“flow control”)
  - avoids slow receivers from being swamped by fast senders

Point-to-Point vs Broadcast

Two categories of networks

- Point-to-point connections
- Broadcast channels
  (also multi-access channels or random access channels)

Many LANs technologies are based on broadcast

➔ Problem: Who gets access to the channel?
Tech speak: Medium access control

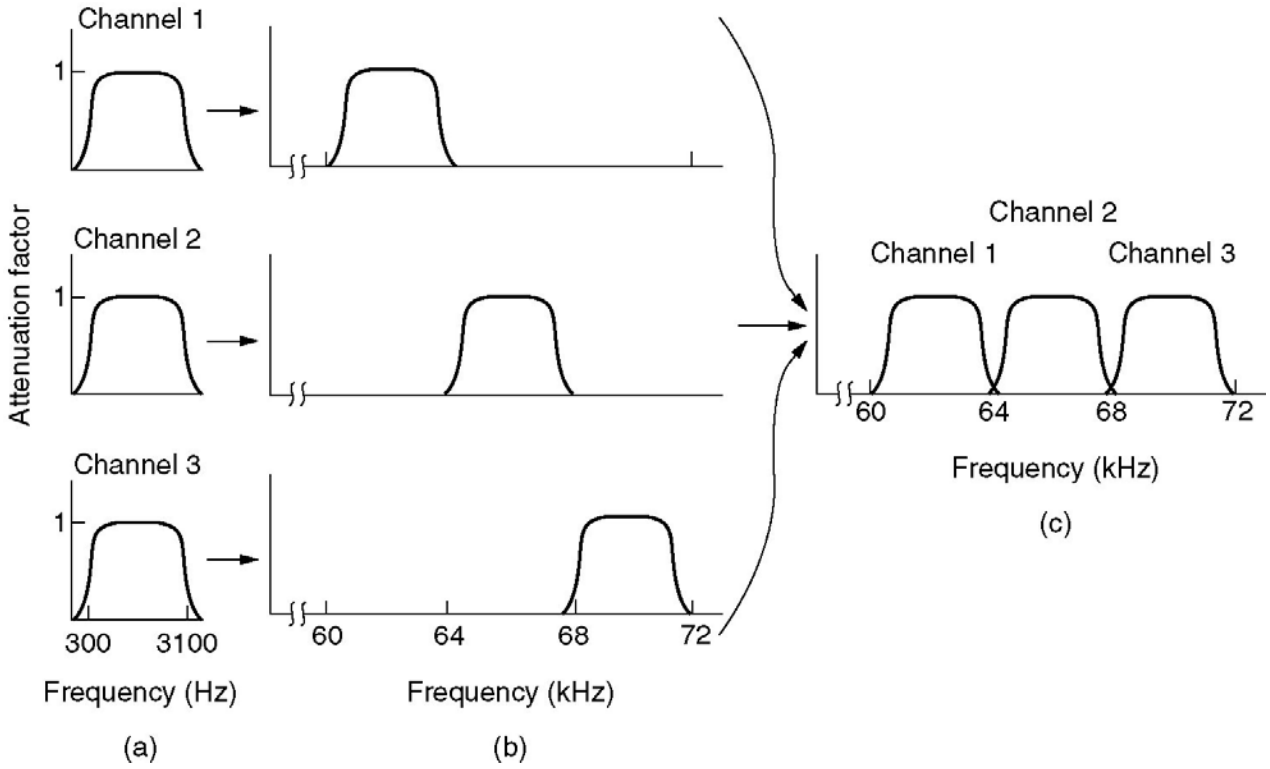
Medium Access Control: Static Approaches

Also called: "Multiplexing"

- Frequency Division Multiple Access (FDMA)
  (telephone trunks, mobiles)
- Time Division Multiple Access (TDMA)
  (trunks, mobiles, Bluetooth, USB)
- Code Division Multiple Access (CDMA)
  (mobile phones, WLANs)

How suitable for computer networks?

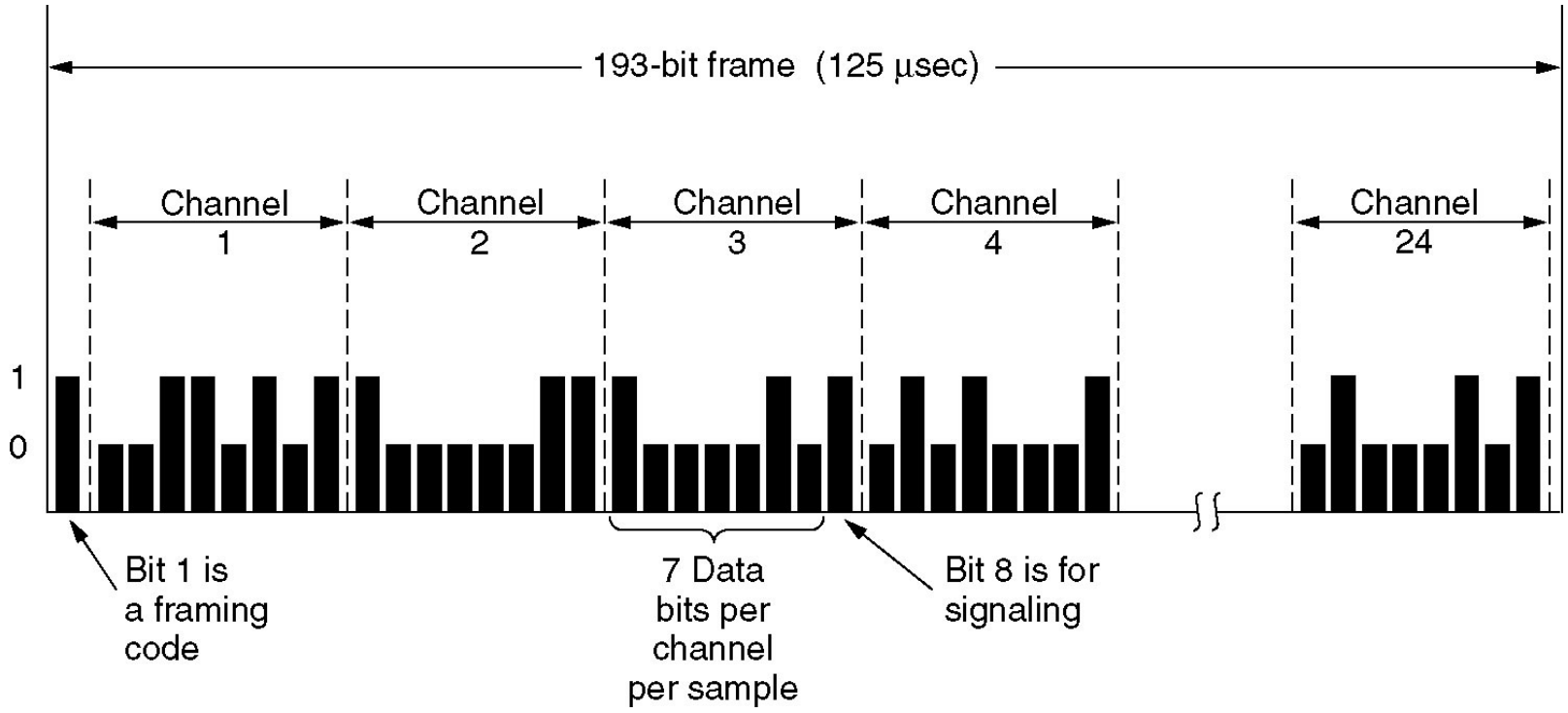
Frequency Division Multiple Access (FDMA)



FDMA on a telephone trunk: (a) original bandwidths
(b) bandwidths raised in frequency (c) multiplexed channel

Time Division Multiple Access (TDMA)

Only applicable to digital data

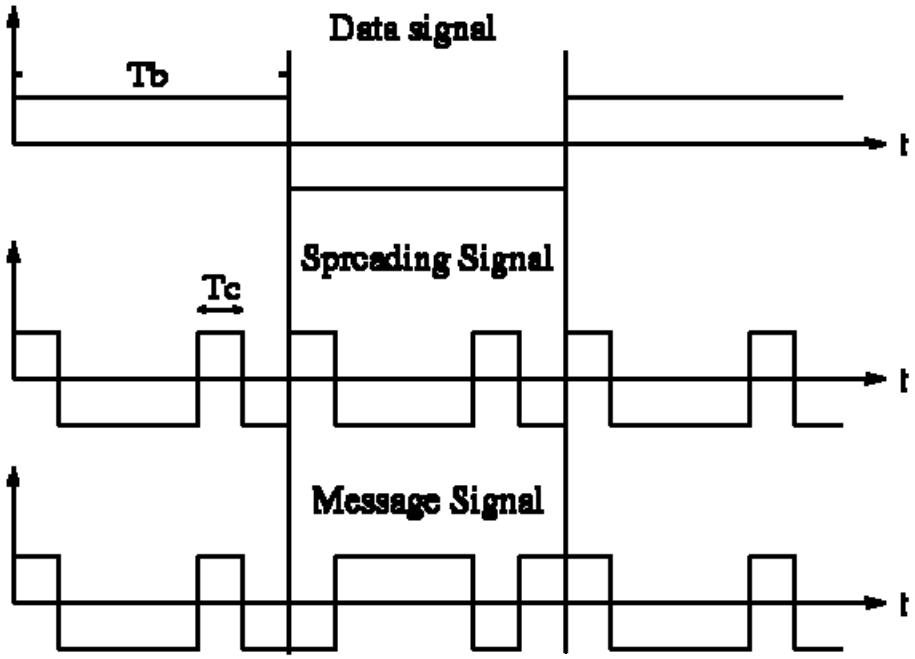


T1 (= Trunk 1) carrier multiplexes 24 voice channels

Gross bandwidth = ?

Code Division Multiple Access (CDMA)

"Spread spectrum" technique



Principle:

- Each station uses a different spreading signal (code)
- Data signal (D) is multiplied by spreading signal (S)
- Mathematics: D_jS_j can be retrieved from $\Sigma\ D_iS_i$ if $(S_i)_i$ are "well chosen"

Medium Access Control: Dynamic Approaches

- ALOHA, slotted ALOHA
- CSMA (Carrier Sensing Multiple Access)
- CSMA/CD (CSMA/Collision Detection)

  original Ethernet
- MACA (Multiple Access with Collision Avoidance)
- MACAW (MACA for Wireless)

Used/proposed for computer networks

ALOHA

Abrahamson, 1970, University of Hawaii

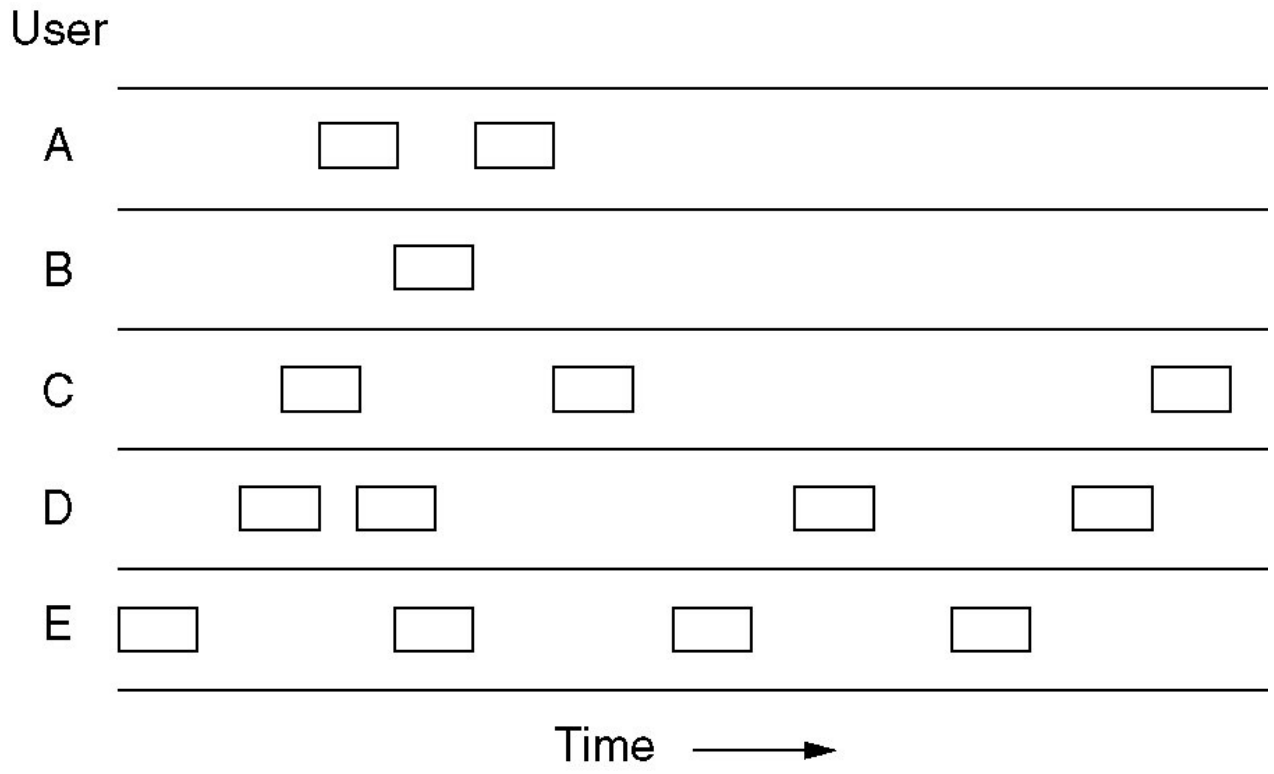
Basic algorithm:

- Stations send frames (which all have the same length) whenever they want
- Then check whether there was a collision with another frame
- In case of a collision, wait a random period and resend

Problem: High probability of collisions

Theory shows: maximal throughput of 18%
for average of 0.5 transmission attempts per frame time

ALOHA



Frames are sent at arbitrary times.

How can one improve upon this?

Slotted ALOHA

Roberts, 1972

Principles like ALOHA, but

- stations can ony send at defined time points (“slots”)
- interval between time points wide enough to fit a frame

Theory shows: maximal throughput of 37%
for average of 1 transmission attempt per frame time

Carrier Sensing Multiple Access (CSMA)

Stations

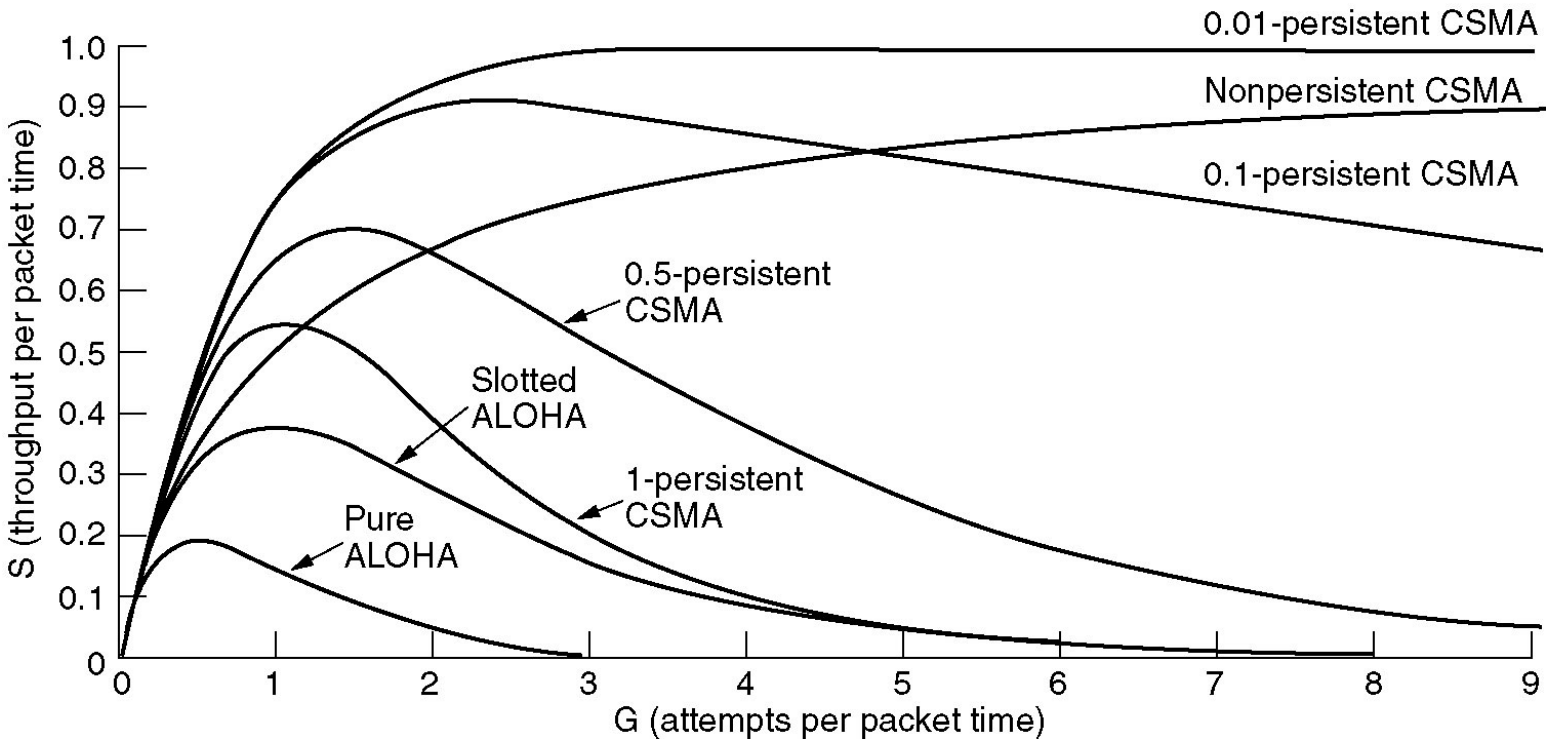
- listen whether there is a “carrier wave” (= carrying data),
- send frame only if channel is idle

3 variants (Kleinrock/Tobagi, 1975)

- persistent:
  - sense and send as soon as channel is idle
- nonpersistent:
  - if channel is busy, wait a random time before sensing again
- p-persistent (0 < p < 1), only defined for slotted channels:
  - with probability p, send!
  - with probability 1-p, defer!

What would you expect for the throughput?

Throughput for ALOHA and CSMA

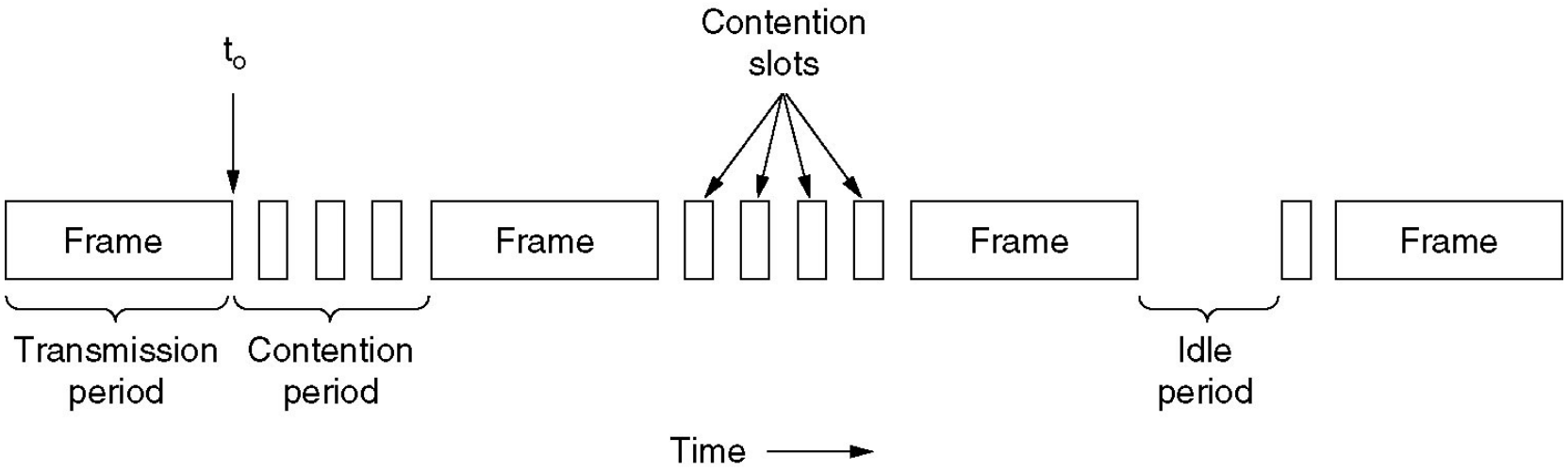


What is the price for good channel utilization?

CSMA with Collision Detection

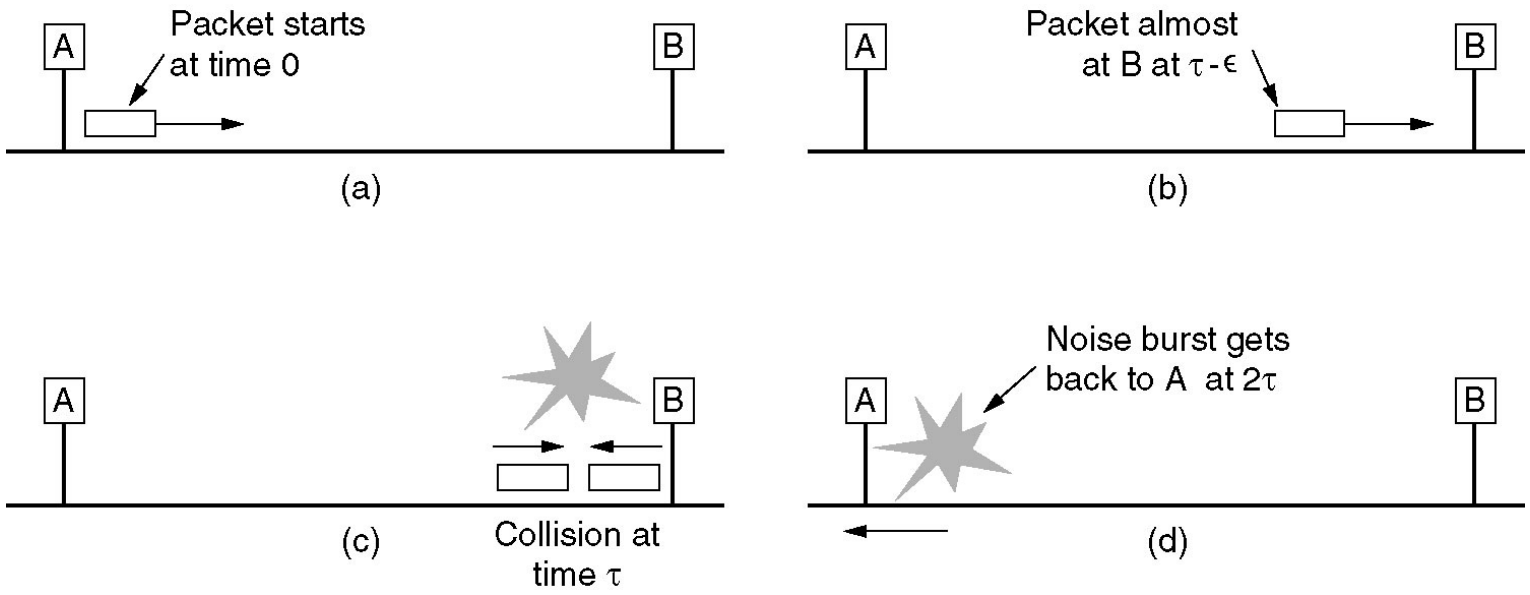
Like persistent CSMA, but stations

- stop sending immediately, when you sense a collision
- wait a random time
- resend



After how much time does a station know that it can send without problem?

Collision Detection



τ is the propagation time $\Rightarrow$ RTT = 2τ

Collision Detection on the Ethernet

Ethernet LAN of 2500 metres:

- What is the RTT if signals travel at no more than 1/3 of the speed of light?

10 Mbps Ethernet:

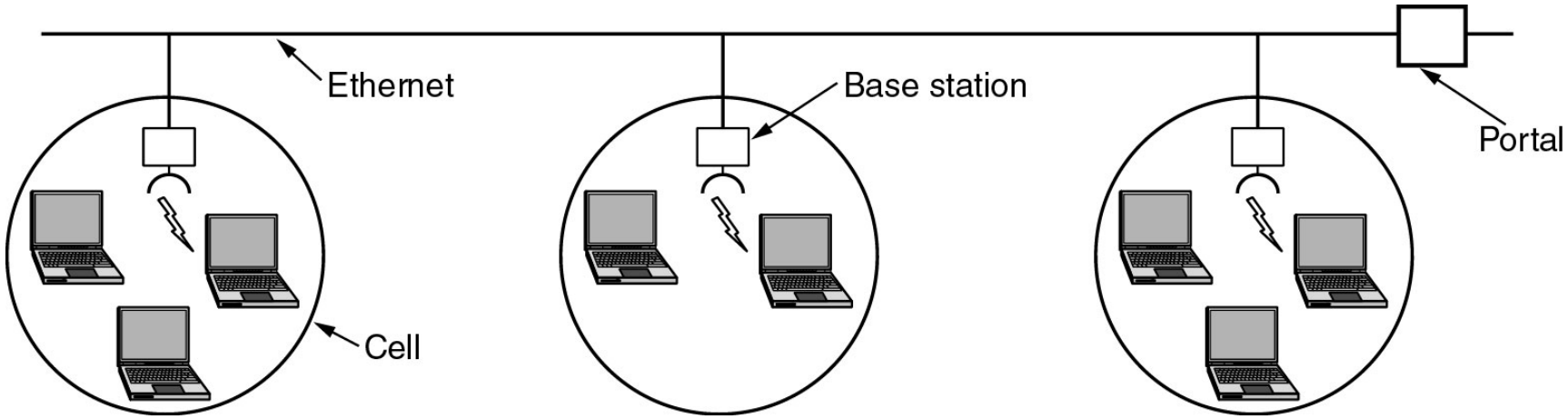
- How many bits can be sent during that time?
- What would be a good minimal length for frames (in bytes)?

Ethernet frames: header has 14 bytes and checksum has 4 bytes

- What should be the minimal payload (in bytes)?
- What should a station do if it has less data to send?

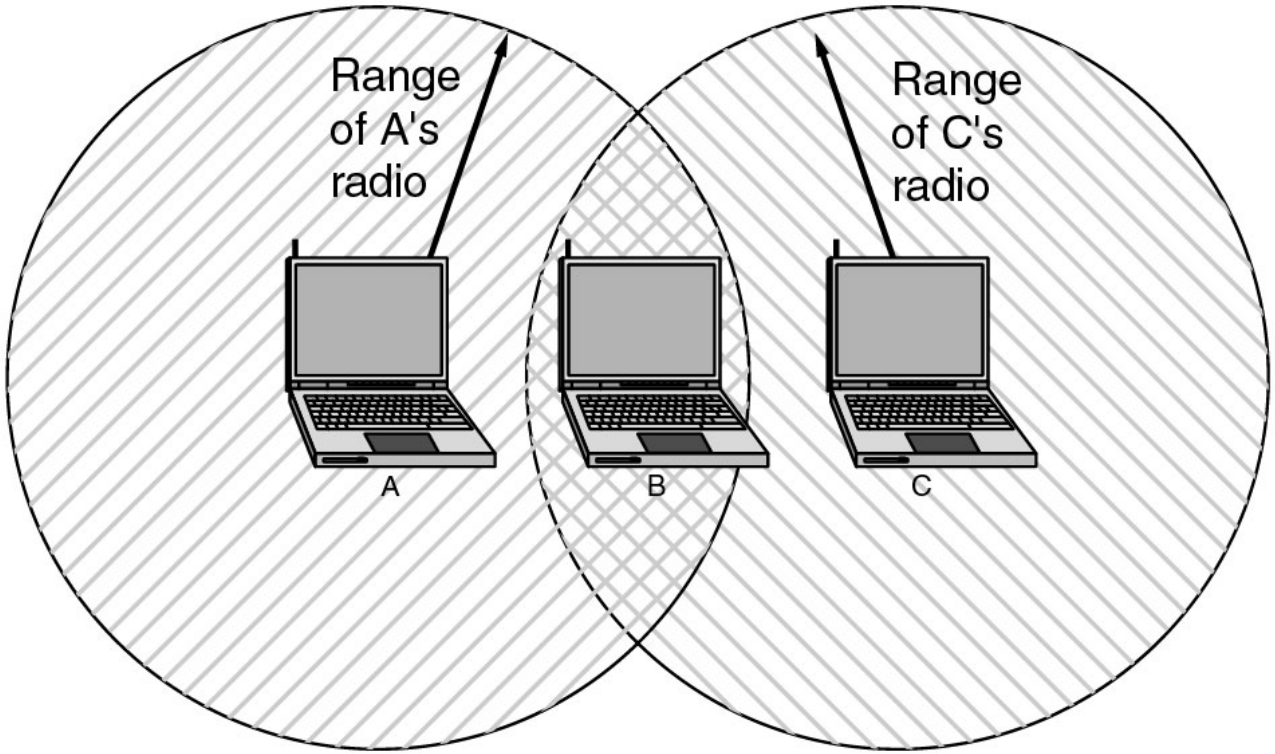
If you do the math right, you arrive at the IEEE 802.3 standard…

A Wireless LAN



Why Don't We Use CSMA/CD?

Do Stations Really Share a Joint Medium?



Hidden and Exposed Stations

Hidden Station Problem:

- If C is sending to B, would A notice?

➔ C is hidden from A

Exposed Station Problem:

- If B is sending to A, could C send to D?

➔ C is exposed to B

Multiple Access Collision Avoidance (MACA)

Karn, 1990

Carrier sensing alone does not work:

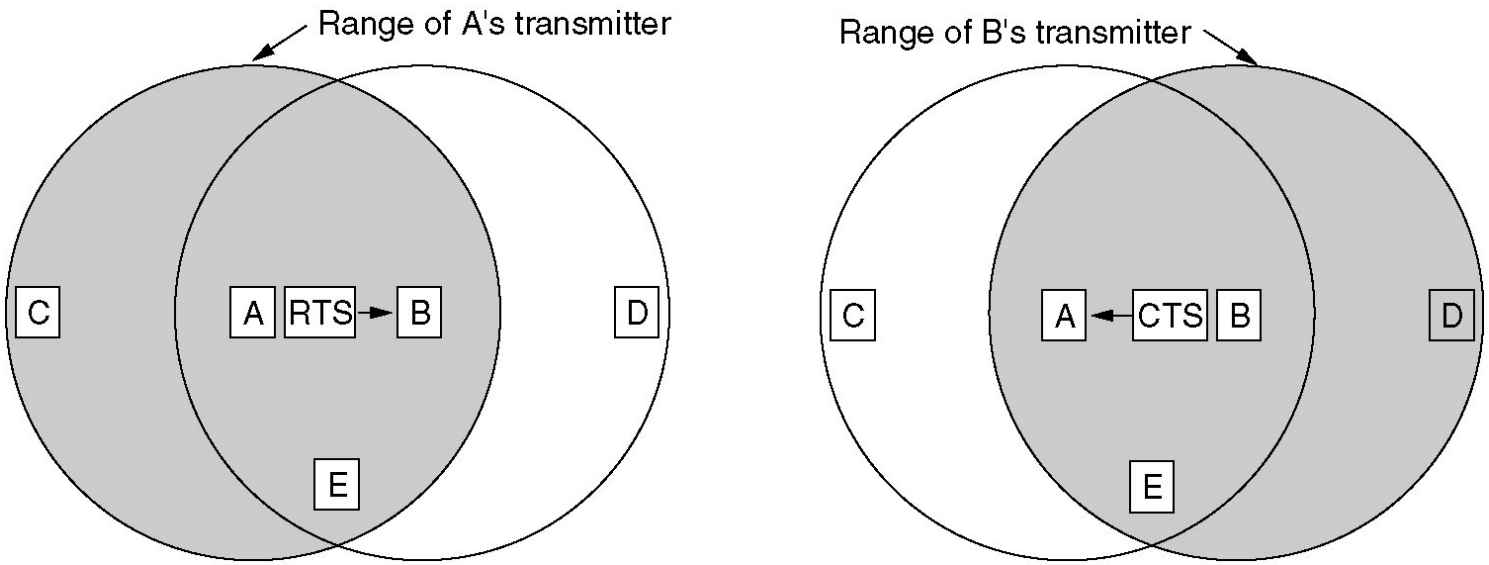
➔ both, sender and receiver, have to agree that “medium” is free:

- Sender outputs a short “probe” frame before the “real” frame
  (RTS = request to send)
- Sender waits for a confirmation by intended receiver
  (CTS = clear to send)
- Only then sender outputs the frame

? What information should RTS and CTS contain?

? How should other stations react?

The MACA Protocol



Does this solve

(a) the hidden station problem? How?

(b) the exposed station problem? Why?

MACA for Wireless (MACAW)

Bhargavan et al., 1994

Answers questions that MACA left open:

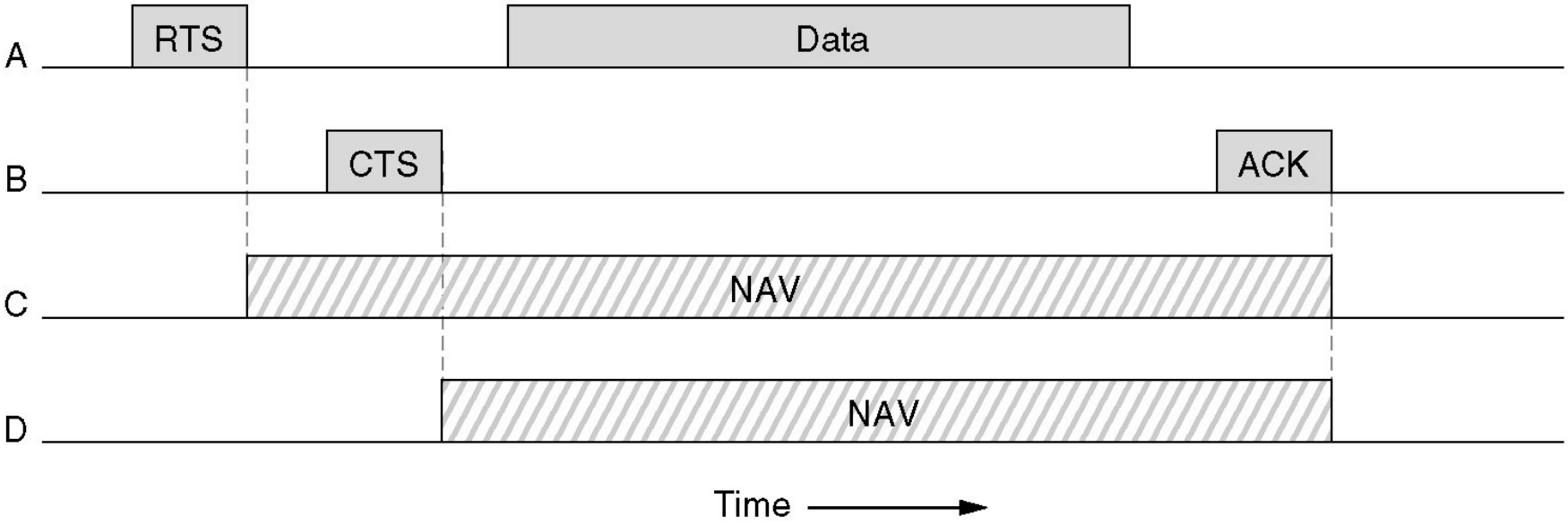
- When does a sender know the frame it sent was corrupted?
  ➔ receiver sends ACK

- What should two senders do if they emit a RTS at the same time?
  ➔ use carrier sensing in the first place, then CA

- What should the “backoff” time depend on?
  ➔ on (sender, receiver)-pair, not sender alone

CSMA with Collision Avoidance (CSMA/CA)

Stations overhearing an RTS or CTS signal create in memory a "virtual signal" (NAV) recording how long the medium will be busy. (NAV = Network Allocation Vector)



Why is D's NAV shorter than C's NAV?

Collision-free Protocols

Collisions during contention period cost time

- can we allocate slots to stations
  without the danger of collision?

  ➔ 2 phases: contention, sending

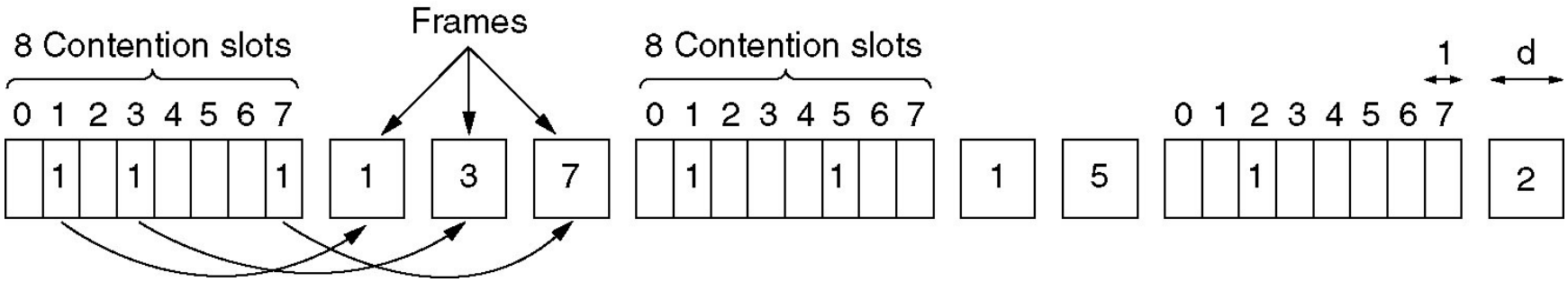
Assumptions:

- stations are numbered 1,...,N
- time is slotted

Not yet taken up by industry

Bit-Map Protocol

- Contention period consists of N 1-bit slots, one for each station
- Reservation: only station i transmits at slot i
  - 1 = frame to send
  - 0 = no frame to send
- Then transmission in numerical order



Pros? Cons?

Binary Countdown: High Numbers Win

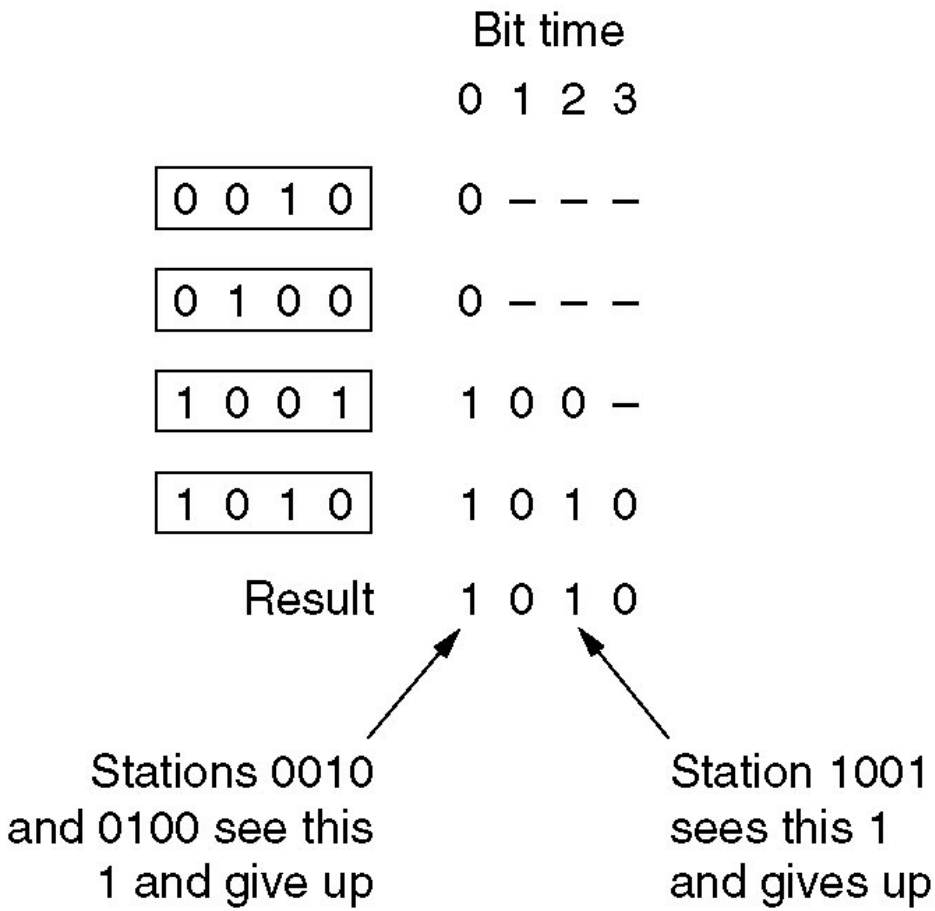
K contention slots where K = ceiling($\log_2$ N)

Example: among 15 stations, consider station S with
number 11 = $(1011)_2$

- 4 contention slots:
  all stations listen and send (or not), signals are ORed
  - 1st slot: S sends 1
  - 2nd slot: S sends 0
  - 3rd slot: if OR of 2nd slot was 0, then S sends 1
    else S is silent
  - 4th slot: ...
- After contention period: everyone knows the number of the top node willing to send

Would this work on a 2.5km long Ethernet?

Binary Countdown: Example



Pros and Cons

	Delay	**Channel Efficiency** (bps successfully transmitted)
Low load	**Contention-based Protocols**	
High load		**Collision Free Protocols**

Ethernet — History

1970, N. Abrahamson, Alohanet, U Hawaii

1975, B. Metcalfe, Ethernet, Xerox Palo Alto Research Centre (PARC)

1978 DIX standard by DEC, Intel, Xerox (10 Mbps)

1982 IEEE 802.3 standard (similar to DIX)

Varieties of Ethernet

10 Mbps

used hubs

Name	Cable	Max. seg.	Nodes/seg.	Advantages
10Base5	Thick coax	500 m	100	Original cable; now obsolete
10Base2	Thin coax	185 m	30	No hub needed
10Base-T	Twisted pair	100 m	1024	Cheapest system
10Base-F	Fiber optics	2000 m	1024	Best between buildings

100 Mbps ("Fast Ethernet")

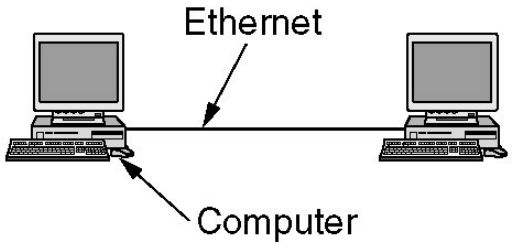
obsolete

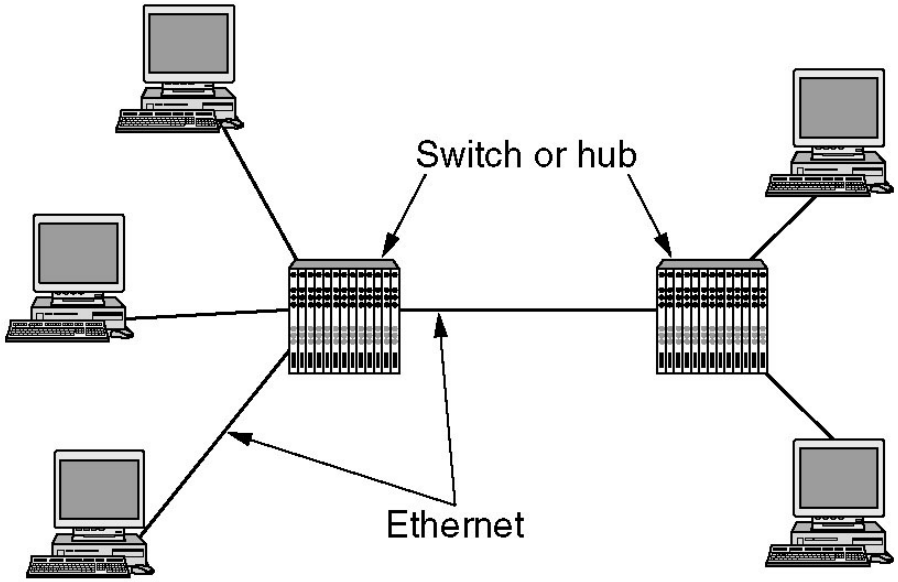
widely used

Name	Cable	Max. segment	Advantages
100Base-T4	Twisted pair	100 m	Uses category 3 UTP
100Base-TX	Twisted pair	100 m	Full duplex at 100 Mbps
100Base-FX	Fiber optics	2000 m	Full duplex at 100 Mbps; long runs

UTP = "Unshielded Twisted Pair"

Varieties of Ethernet — Gigabit Ethernet





Name	Cable	Max. segment	Advantages
1000Base-SX	Fiber optics	550 m	Multimode fiber (50, 62.5 microns)
1000Base-LX	Fiber optics	5000 m	Single (10 μ) or multimode (50, 62.5 μ)
1000Base-CX	2 Pairs of STP	25 m	Shielded twisted pair
1000Base-T	4 Pairs of UTP	100 m	Standard category 5 UTP

MAC Addresses

Each network interface in the world has a unique address: MAC address

Format: 2 x 3 bytes

- 3 bytes Organisationally Unique Identifier (= vendor specific)
- 3 bytes Network Interface Controller (NIC) specific

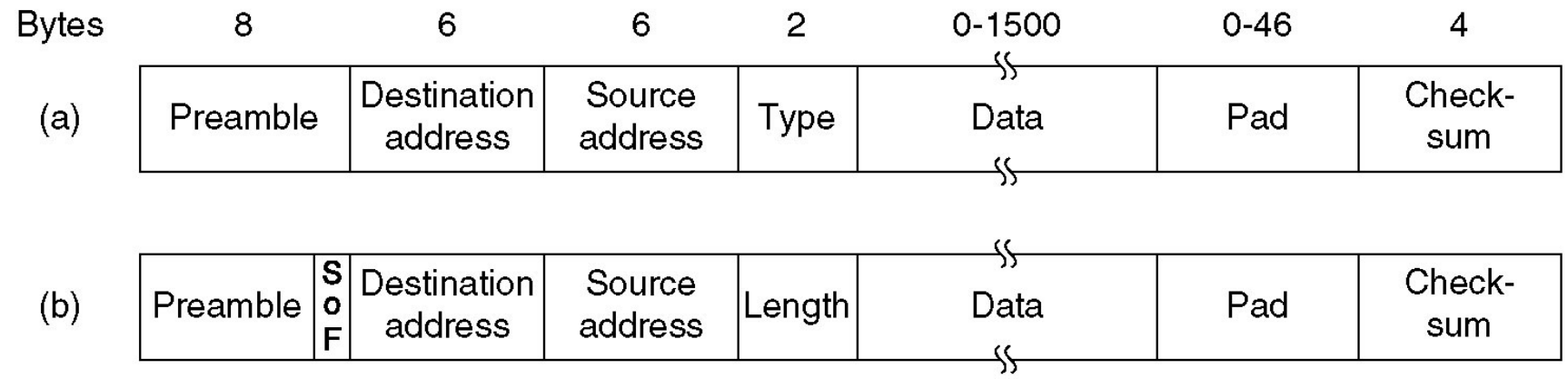
Examples:

- 00:16:cb:c9:a2:2c *(Ethernet address of my MAC)*
- ff:ff:ff:ff:ff:ff (broadcast address)
- Group addresses have 8th bit = 1, individual addresses have 8th bit = 0

Other technologies use MAC addresses too, for example:

- 802.11 wireless network, Bluetooth, IEEE 802.5 token ring (and most other IEEE 802 networks), FDDI (glass fibre)

Ethernet Frames



(a) IEEE 802.3

(b) DIX Ethernet

Preamble: 1010101010, eight times (used for synchronization)

SoF: 1010101011 (start of frame delimiter)

Type: Type of payload (e.g., 0800_{16} stands for IP)

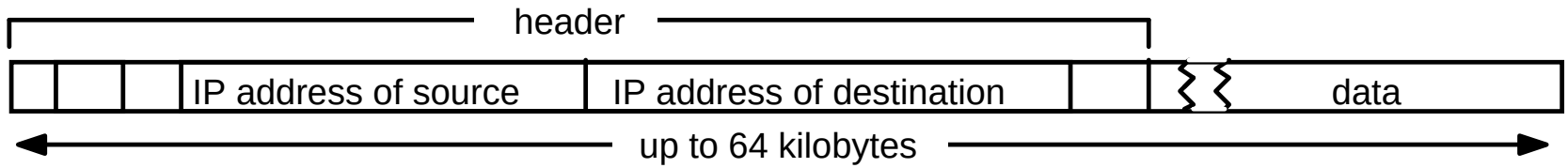
2. Networking

2.4 Network Layer

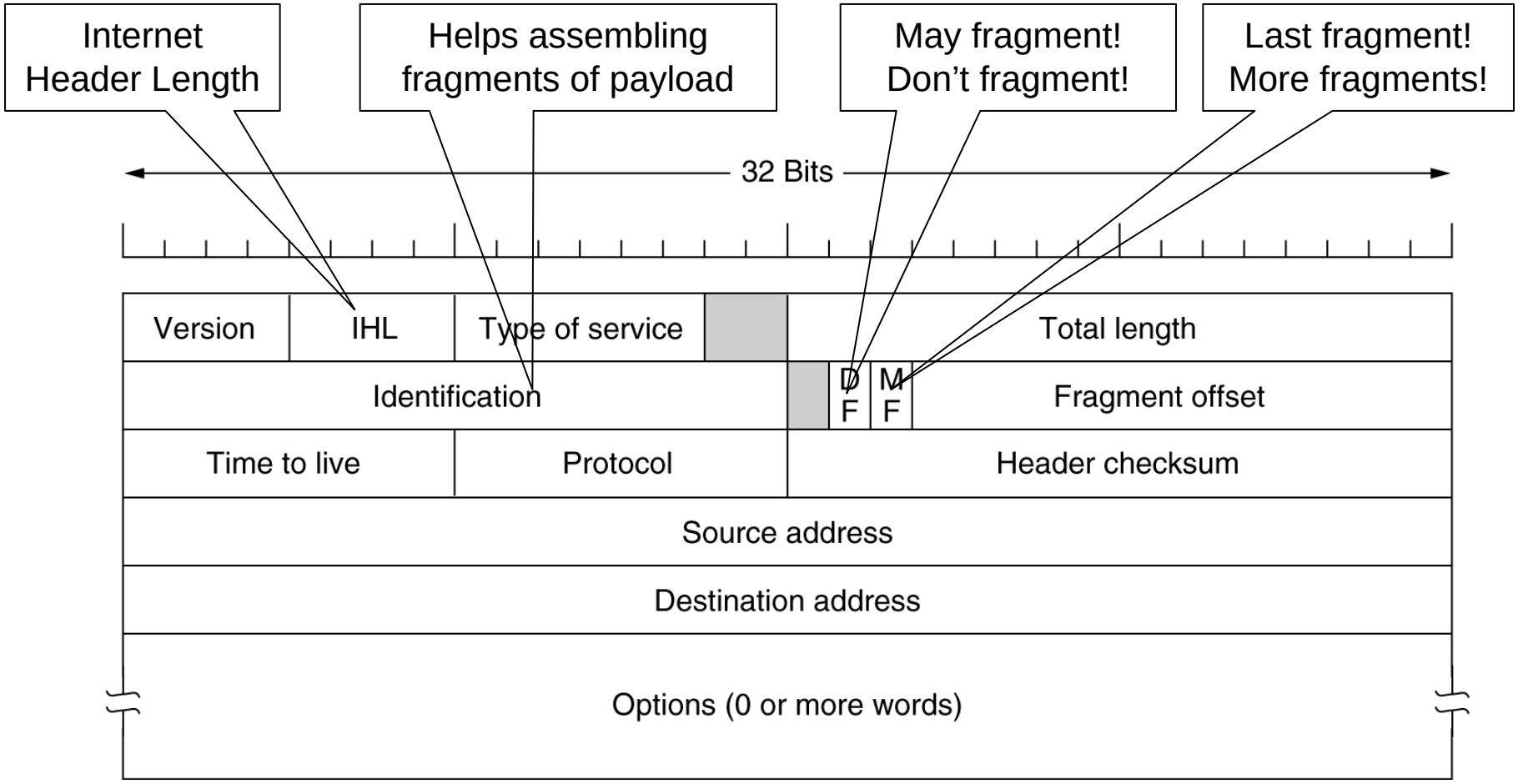
Internet Protocol (IP)

- Enables hosts to send packets to other hosts

- Layout of an IP packet



IP Packet Structure



IP version 4 (IPv4) header

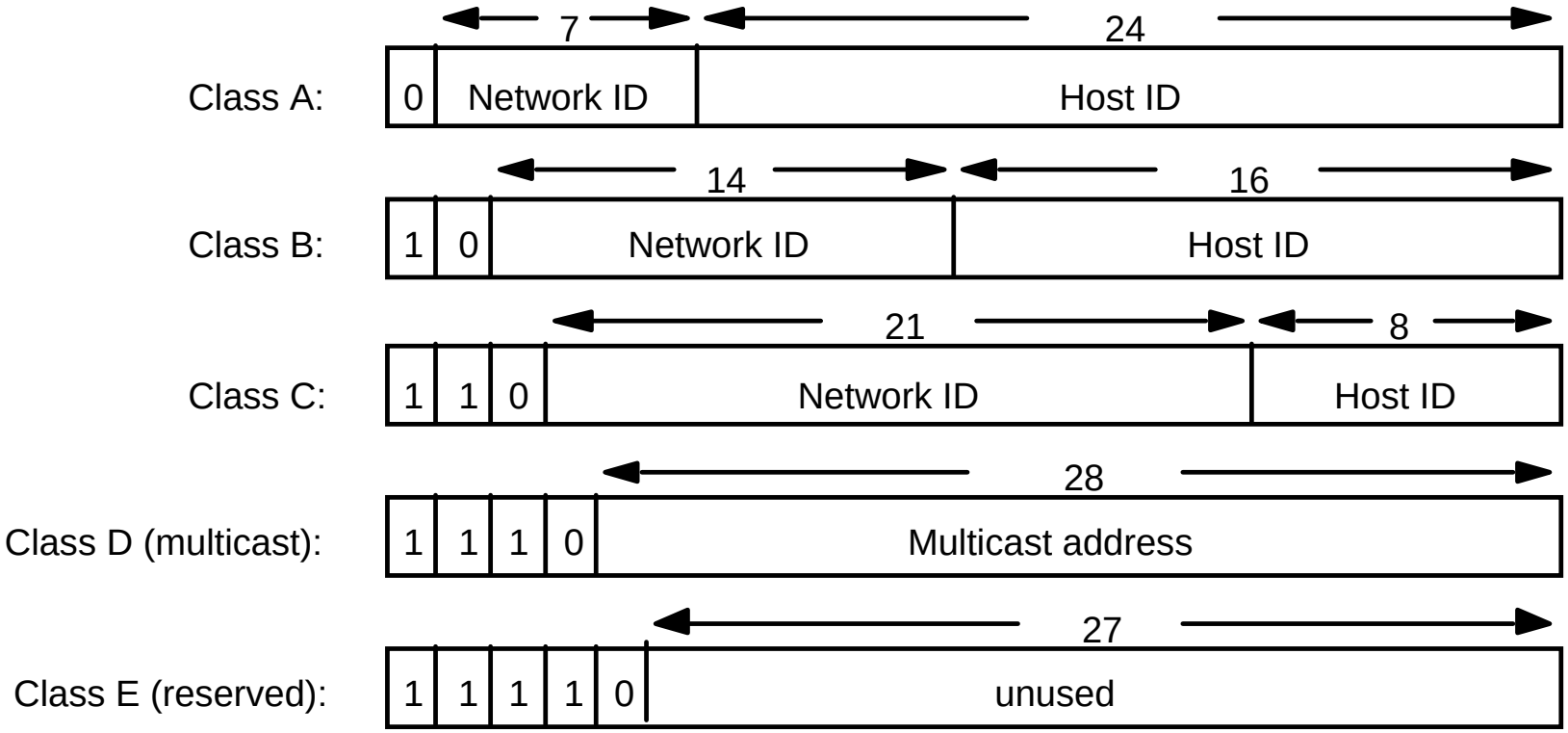
IP Addresses

Address format:

- 32 bits = 4 bytes (octets)
- Representation in “dotted decimal” notation
  193.206.186.140
- Representation in hexadecimal code
  0xc1ceba8c
- Representation in bit code
  11000001 11001110 10111010 10001100

How can packets be routed, based on their address?

IP Addresses



Originally, IP addresses were divided into classes …

Hosts belong to Networks, Addresses Belong to Network Ranges

MIT Network	18.0.0.0 - 18.255.255.255
Unibz Network	193.206.186.0 - 193.206.186.255
Yahoo Network	69.147.64.0 - 69.147.127.255

- How do we describe network ranges?

- Note: all addresses in a range
  - agree on their first N bits (network prefix)
  - vary on the remaining 32-N bits (host address)

- CIDR Notation (CIDR = Classless Interdomain Routing)

  - MIT Network 18.0.0.0/8
  - Unibz Network 193.206.186/24
  - Yahoo Network 69.147.64.0/18

Subnet Masks

Suppose: A packet for destination 193.206.186.140 arrives at router

How does the router know to which network the packet should go?

Routers have two pieces of information per network entry

- Network address 193.206.186.0
- 32 bit mask 255.255.255.0

Algorithm:

For each network entry

compute: (destination address) AND (subnet mask)

if result = network address, then destination in network

Special Addresses

- Address ranges for private networks
  (assumed to have no contact to the Internet!?):

  10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16

- Network address: lowest number in range
- Broadcast address: highest number in range
- Gateway address: often second highest number in range
- Loopback network: 127.0.0.0/8
  virtual interface connection a host to itself
- Localhost: 127.0.0.1

IP Address Quiz

- How many possible subnet masks are there?
- What are the possible numbers that can occur in a mask position?
- What is the network mask of the Stanford Univ. network (171.64.0.0/14)?
- How many addresses are there on the Stanford network?
- Which of the following addresses could belong to a host at Stanford:
  171.74.212.31 ?
  171.68.0.31 ?
  171.67.212.44 ?
- Host **`actarus.inf.unibz.it`** has the address 10.10.20.5 and mask 255.255.252.0.

  What is the broadcast address on that host's network?
  What is (probably) the gateway address?

Routing Information on a Host

Every IP capable host needs to know about at least two classes of destinations

- locally connected computers
- everywhere else

Routing table on actarus:

```
wnutt@actarus:~$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
10.10.20.0      *               255.255.252.0   U         0 0          0 eth0
10.10.112.0     *               255.255.240.0   U         0 0          0 eth1
default         10.10.23.254    0.0.0.0         UG        0 0          0 eth0
```

Address Resolution

Running ARP (= Address Resolution Protocol),
a host finds out the MAC address belonging to an IP address

ARP Steps

- actarus wants to send an IP packet to 10.10.23.254
- actarus broadcasts an ARP request:
  "I have IP address 10.10.20.5 and MAC address 00:11:85:e8:ff:8f, who has IP address 10.10.23.254?"
- Gateway sends an ARP reply to actarus:
  "I have IP address 10.10.23.254 and MAC address 00:10:db:bd:ce:87"

Optimization

- Hosts keep a cache
- Hosts overhearing a request update their cache
- ARP announcements: hosts send an ARP request to themselves (why?)

IP Routing (1)

Problem: Host H1 wants to send a packet to host H2

Case 1: H2 is on the same LAN (e.g., Ethernet) as H1

Approach:

- H1 finds out the Ethernet address of H2 (MAC address)

  (physical address, unique in the world
  for every Ethernet-enabled device)

- Ethernet module of H1 sends out the packet
  in Ethernet format

IP Routing (2)

Case 2: H2 is on a different LAN

Approach:

- H1 sends packet to its local gateway (say, G1)
- G1 sends packet across intermediate networks to the network of H2

If a gateway receives a packet, where should it send it?

Routing Problem

- What is a good path from H1 to H2?
- What is the next step on the path?

Computers forwarding packets through a network are called routers

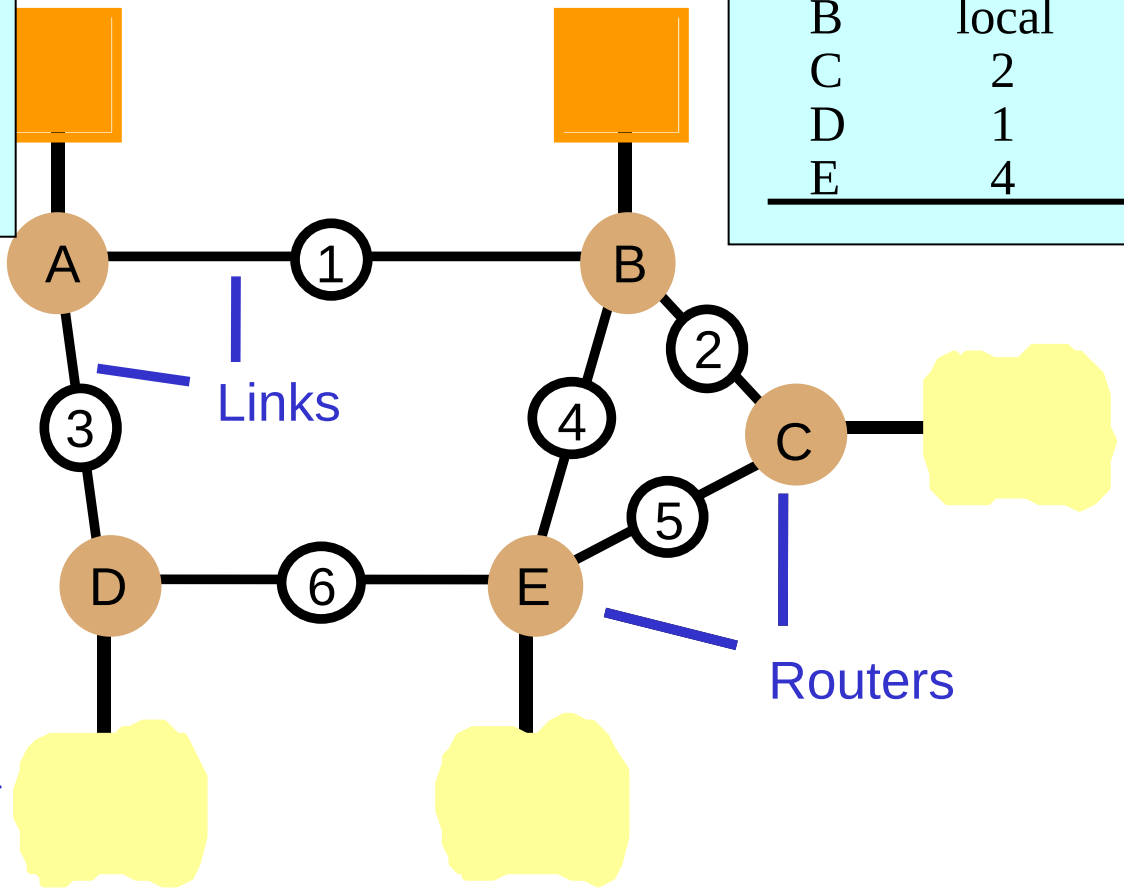
Routing: Example

Routings from A

To	*Link*	*Cost*
A	local	0
B	1	1
C	1	2
D	3	1
E	1	2

Routings from B

To	*Link*	*Cost*
A	1	1
B	local	0
C	2	1
D	1	2
E	4	1



Routing Tables

Routings from A

To	*Link*	*Cost*
A	local	0
B	1	1
C	1	2
D	3	1
E	1	2

Routings from B

To	*Link*	*Cost*
A	1	1
B	local	0
C	2	1
D	1	2
E	4	1

Routings from C

To	*Link*	*Cost*
A	2	2
B	2	1
C	local	0
D	5	2
E	5	1

Routings from D

To	*Link*	*Cost*
A	3	1
B	3	2
C	6	2
D	local	0
E	6	1

Routings from E

To	*Link*	*Cost*
A	4	2
B	4	1
C	5	1
D	6	1
E	local	0

Sample Routes

- Send from C to A:
  - to link 2, arrive at B
  - to link 1, arrive at A

- Send from C to A if B's table is modified to:

Routings from B

To	*Link*	*Cost*
B	local	0
C	2	1
E	4	1
default	5	-

  - to link 5, arrive at E
  - to link 4, arrive at B
  - to link 1, arrive at A

Note the extra hop.

Approaches to Routing Algorithms

Decentralised

- a router communicates with its immediate neighbours

➔Distance Vector algorithm (Bellman, Ford, Fulkerson)

  – realised in Router Information Protocol (RIP)

Global

- a router knows all routers in the network, their links, and the cost of sending a packet over a link

  (also called: link state protocols)

➔ Shortest Path algorithm (Dijkstra),

  – realised in Open Shortest Path First (OSPF) protocol

Distance Vector Routing: Principles

- Each router R maintains a routing table *(= distance vector)*, which records for each other router how far away it is from R *(e.g., how many hops)*
- The initial table of R has only one element: (R,local,0)
- Periodically, or when there is a change in its neighbourhood, a router sends its table to its neighbours
- When receiving a table, a router updates its local table
- When a link to a neighbour fails,
  the cost of the link is set to ∞

How does a router know that a link has failed?

Distance Vector Algorithm: Idea

Update: Every *t* seconds or when local table changes,
send the full table to each accessible neighbour.

Propagation: When receiving an update from neighbour N

- if N knows a path to a new destination D,
  send messages for D to N
- if N knows a cheaper path to D, send messages for D to N
- if N is closer to D *(i.e., messages for D are sent to N)*,
  update cost for D
  (Idea: N has better information about D)

See next slide for details

Distance Vector Algorithm (Pseudo Code)

Send: Every *t* seconds or when local table *Tl* changes,
send *Tl* on each non-faulty outgoing link

Receive: Whenever a routing table *Tr* is received on link *n*:

```
for all rows Rr in Tr {// modify Rr for subsequent comparisons
    if (Rr.link ≠ n) {
        Rr.cost = Rr.cost + 1;
        Rr.link = n;
        if (Rr.destination is not in Tl) add Rr to Tl;
         // add new destination to Tl
        else for all rows Rl in Tl {
            if (Rr.destination = Rl.destination and
               (Rr.cost < Rl.cost or Rl.link = n)) Rl = Rr;
            // Rr.cost < Rl.cost : remote node has better route
            // Rl.link = n : remote node is more authoritative
        }
    }
}
```

Distance Vector Routing: Convergence

- After initialisation, all routers reach a state where all tables are correct
  (i.e., show next hop along shortest path)
- Similarly, after a new router has joined
- However, convergence is slow

Distance Vector Routing: Looping

- When links fail, tables may be updated in a way that leads to loops

  rare situation, caused by delayed messages

- Routers in a loop continuously update their tables, increasing the cost ("count to infinity")

- Solution (among others): make infinity small

  RIP: $\infty = 16$

Distance Vector Routing: Protocols

- RIP was the first Internet routing protocol
- Not scalable
- Replaced by a link state protocol

Link State Routing: Principles

- A router knows its neighbourhood, i.e.,
  - the routers it is linked to
  - the cost of the links
- Periodically, it broadcasts a map of its neighbourhood
  (the neighbourhood maps have timestamps)
- Each router
  - builds a global map, using the latest neighbourhood maps
  - computes the shortest path to each other router
- Routing table:
  - for each R, show the first hop on the shortest path to R

Dijkstra's Algorithm (1)

Input:

- graph G = (V,E)
- weight function w: E $\rightarrow$ R
- start node s $\in$ V

Output:

- function d: V $\rightarrow$ R
  - v.d is the distance from s to v (= length of shortest path)
- function pred: V \ {s} $\rightarrow$ V
  - v.pred is the predecessor of v on the shortest path from s to v

Dijkstra's Algorithm (2)

Input: $G = (V,E)$, $w: E \to R$, $s \in V$

Output: $d: V \to R$, $pred: V \setminus \{s\} \to V$

Ideas:

- Initial pessimistic estimates:
  - $v.d = \infty$ for all $v \in V$
  - $v.pred = null$
- Loop:
  - improve estimate of d
  - find candidate for pred
  - determine vertex v such that v.d is exact (and also v.pred)

Dijkstra's Algorithm (Pseudo Code)

```
Input: V, E, w, s

S = ∅, Q = V;          // Initialisation

For each vertex v ∈ V {
    v.d = ∞;
    v.pred = null }
s.d = 0;
While Q is not empty {     // Algorithm
    u = extractMin(Q);      // extract a vertex u for which
                            // u.d is minimal
    S = S ∪ {u};
    For each edge (u,v) outgoing from u {
        if (u.d + w(u,v) < v.d) {   // Relax v.d
            v.d = u.d + w(u,v);
            v.pred = u}
        }
    }
```

Dijkstra's Algorithm: Discussion

- If u = extractMin(Q), then the estimates for u are correct
- Shortest path from s to v:
  follow pred links
- Runtime
  - each vertex and each edge are visited only once
    ➔ total runtime = O(|E| + |V| x runtime(extractMin))
  - runtime of extractMin depends on implementation:
    O(log V) possible
    ➔ total runtime = O((|E| + |V|) x log(V))
- Incremental versions: needed to update routing tables

Routing: How Can All this Work?

The Internet is too large to be captured in one routing table

➔ Divide and Conquer

The Internet is divided into Autonomous Systems (ASs)
(= network with common routing protocol, e.g., RIP or OSPF)

Hierarchical Routing

- Granularity of Internet routing = ASs
- Internal traffic of an AS: finegrained routing
- Outbound traffic: send to (suitable) gateway
- At AS level: apply Boundary Gateway Protocol (BGP)
- Inbound traffic = internal traffic

Which Route Do My Packets Take?

- Unix/Linux: **`traceroute`**
- Windows: **`tracert`**

Example: **`tracert www.yahoo.com`**

How does it work?

- A packet has a time to live (TTL)
  Initially: TTL = 64 hops
- If a packet dies (TTL = 0 hops), most routers send error message back to source (ICMP “time exceeded” packet)
- Iteratively, send packets with TTL = 1, TTL = 2, …

2. Networking

2.5 Transport Layer

UDP and TCP

UDP = User Datagram Protocol

TCP = Transport Control Protocol

- Communication facilities for application programmes
- Implemented on top of IP
- Support communication between ports at hosts

Paradigms

- UDP: Datagram (= short instantaneous message)
- TCP: Byte stream (= arbitrarily long,
  no duplicates, no losses)

Applications and Transport Protocols

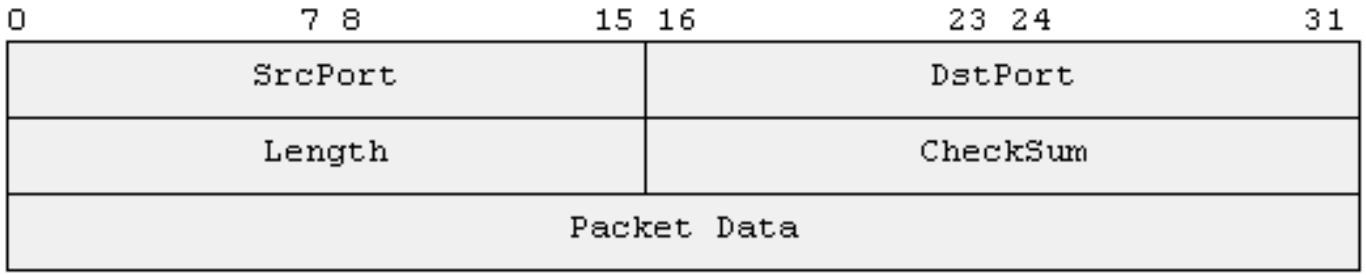
Application	Application-layer protocol	Underlying Transport Protocol
Electronic mail	SMTP	TCP
Remote terminal access	Telnet	TCP
Web	HTTP	TCP
File transfer	FTP	TCP
Streaming multimedia	proprietary	typically UDP
Internet telephony	proprietary	typically UDP
Network management	SNMP	typically UDP
Routing protocol	RIP, OSPF	typically UDP
Name translation	DNS	typically UDP

UDP

- Messages no larger than IP packets
- Header contains
  - source and destination port numbers
  - length of message (at most 64 kbytes)
  - checksum (optional)

- Pro: almost no overhead
- Con: no reliability

UDP Packet Structure



TCP

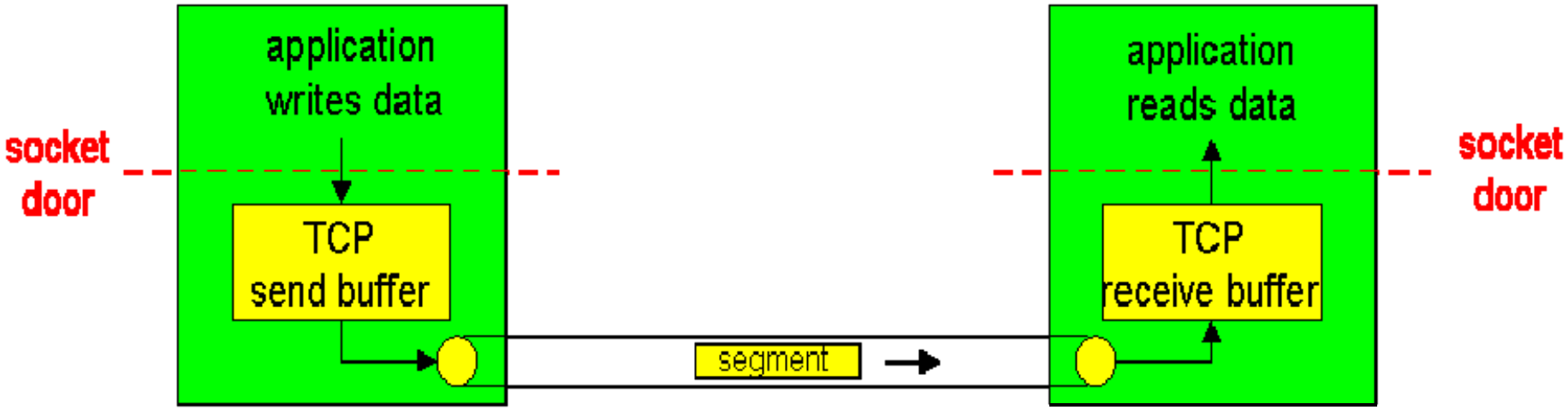
Used by: FTP, Telnet, HTTP, DNS, SMTP, POP3,…

Principle: Two processes communicate via two byte streams

Features:

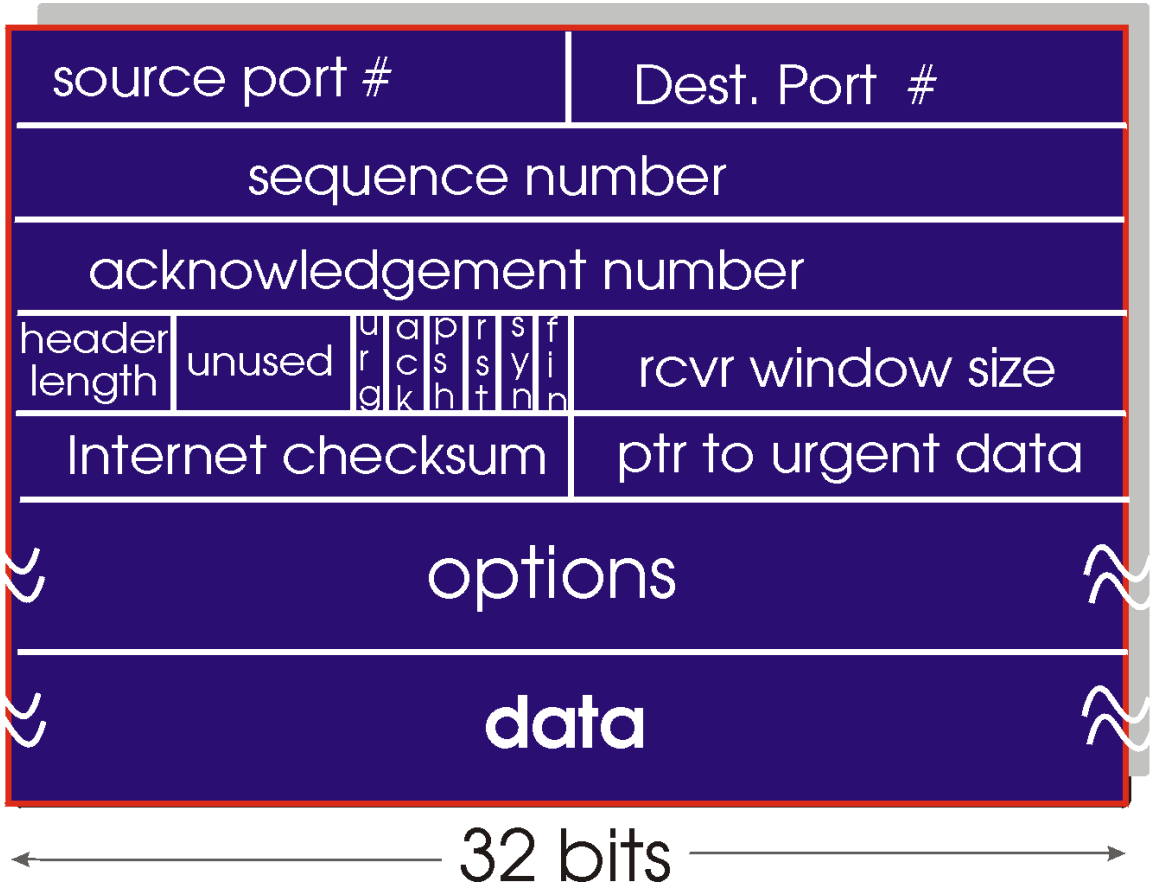
- Virtual connection: runs only on IP
- Point-to-point: single sender and receiver, no multicasting
- Flow control: receiver tells sender how much data it can process
- Reliability: no data are lost (at TCP level)
- Congestion control: sender tries to avoid network congestion

TCP Send and Receive Buffers



- Byte stream is chopped into segments
  (no larger than MSS = maximum segment size)
- Stream segments are packed into “TCP segments”
  (= data + header)
  and handed to IP
- Connection = buffers + state variables

TCP Segment Structure



sequence number, acknowledgement number: reliability

window size:
flow control

Flags:

- syn: request to establish connection
- ack: agreement to connect
- rst: reset connection
- fin: finish connection

Sequence Numbers

Sequence number of a segment:

Byte stream number of first byte in segment

Example: A sends to B over TCP

- 500k image with MSS = 1k,
  initial sequence number = 0

➔ 500 segments,
with sequence numbers 0, 1024, 2048, ...

Acknowledgement Numbers

Acknowledgement number in segment sent from B to A:

Sequence number of next byte B is expecting from A

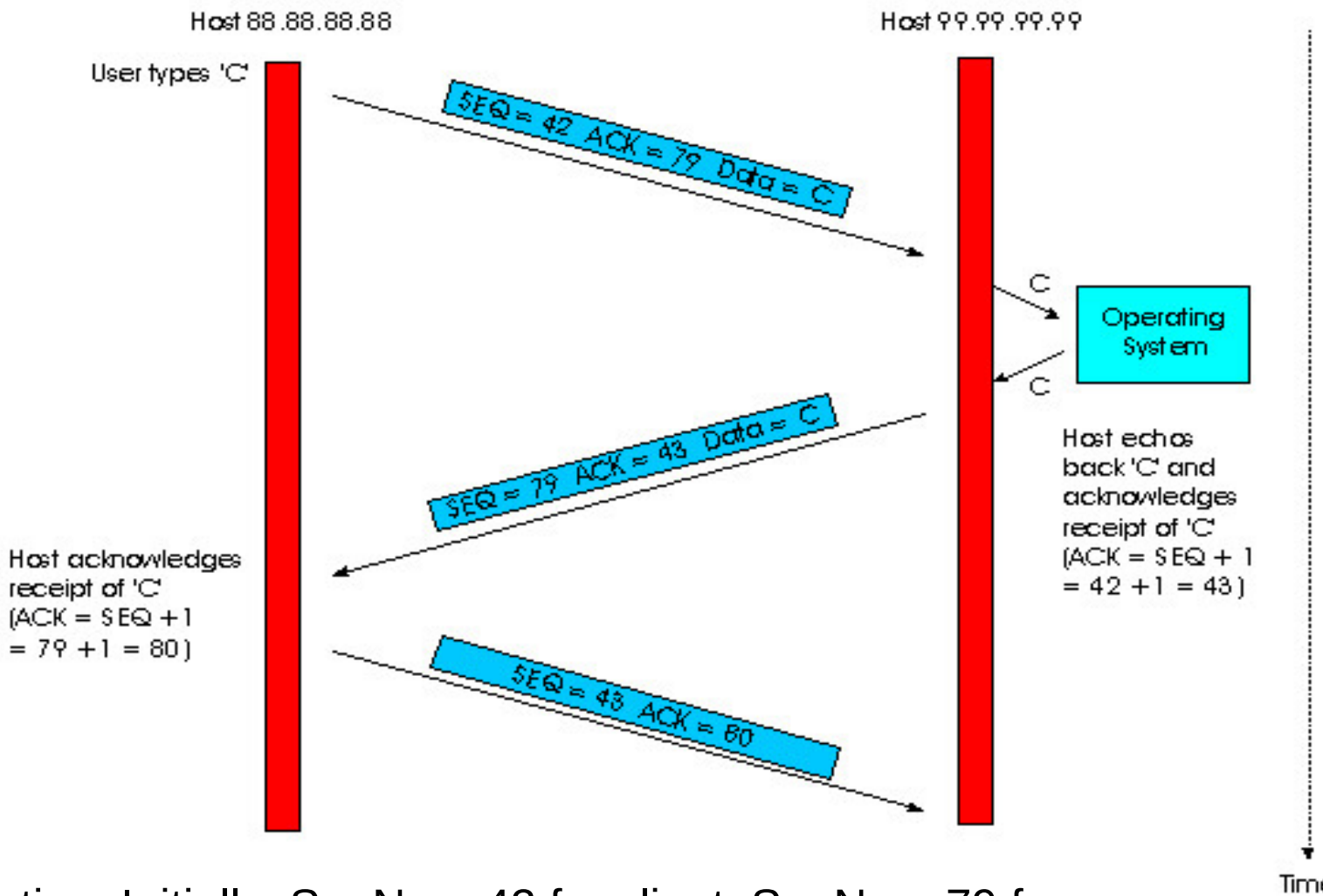
Example:

- B has received segments 1, 2, and 4, but not 3.
- Acknowledgement number is 2048
  (= 1st byte of segment 3)

Example shows:

- Acknowledgement is cumulative
  (acknowledges all bytes up to Ack - 1)
- No mention of out-of-order segments

Example: Sequence and Acknowledgement Numbers in Telnet Application



Assumption: Initially, SeqNo = 42 for client, SeqNo = 79 for server

TCP Sender Actions

Client variables

```
ackSNo  = initialSequenceNumber  // ack'ed sequence number
nextSNo = initialSequenceNumber // next sequence number
```

Loop through the following cases:

```
if (data received from application){
    create segment with sequence number nextSNo;
    start timer for segment nextSNo;
    pass segment to IP;
    nextSNo = nextSNo + data.length}

if (timeout for segment with sNo y){
    retransmit segment y;
    restart timer for segment y}
```

TCP Sender Actions (2)

```
if (ACK received with AckNo = y)
    if (y > ackSNo){ // cumulative ack
        cancel timers for segments with lower SNos;
        ackSNo = y}
    else { // duplicate ack
        increment counter for duplicate acks for y;
        if (number of duplicate acks for y == 3) {
            retransmit segment y;
            restart timer for segment y
            }
          }
```

Sender Actions Ensure Reliable Data Transfer

Timers for all segments

Retransmit of segment y occurs if

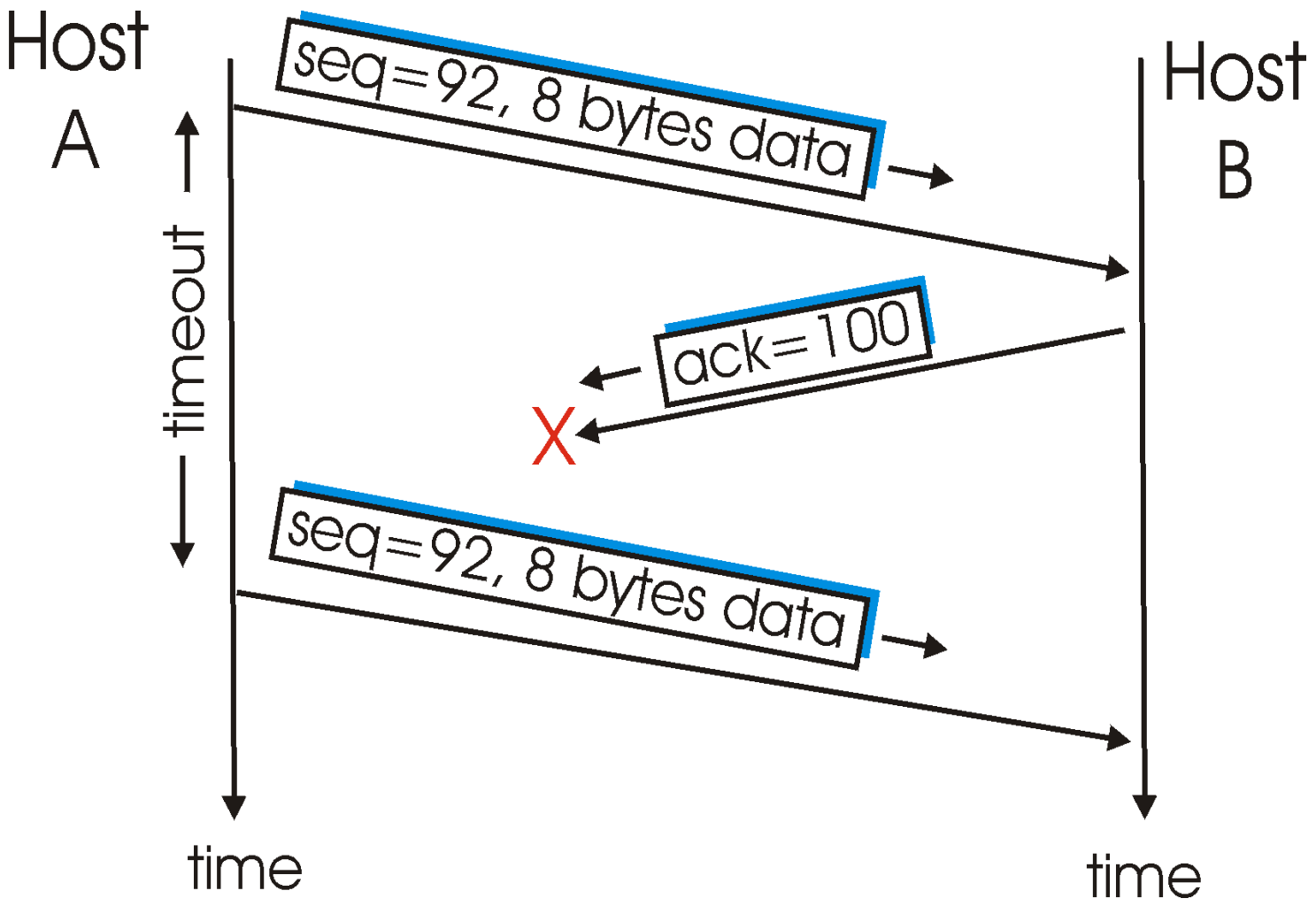
- timer has timed out, or
- three duplicate acks have been received

Why?

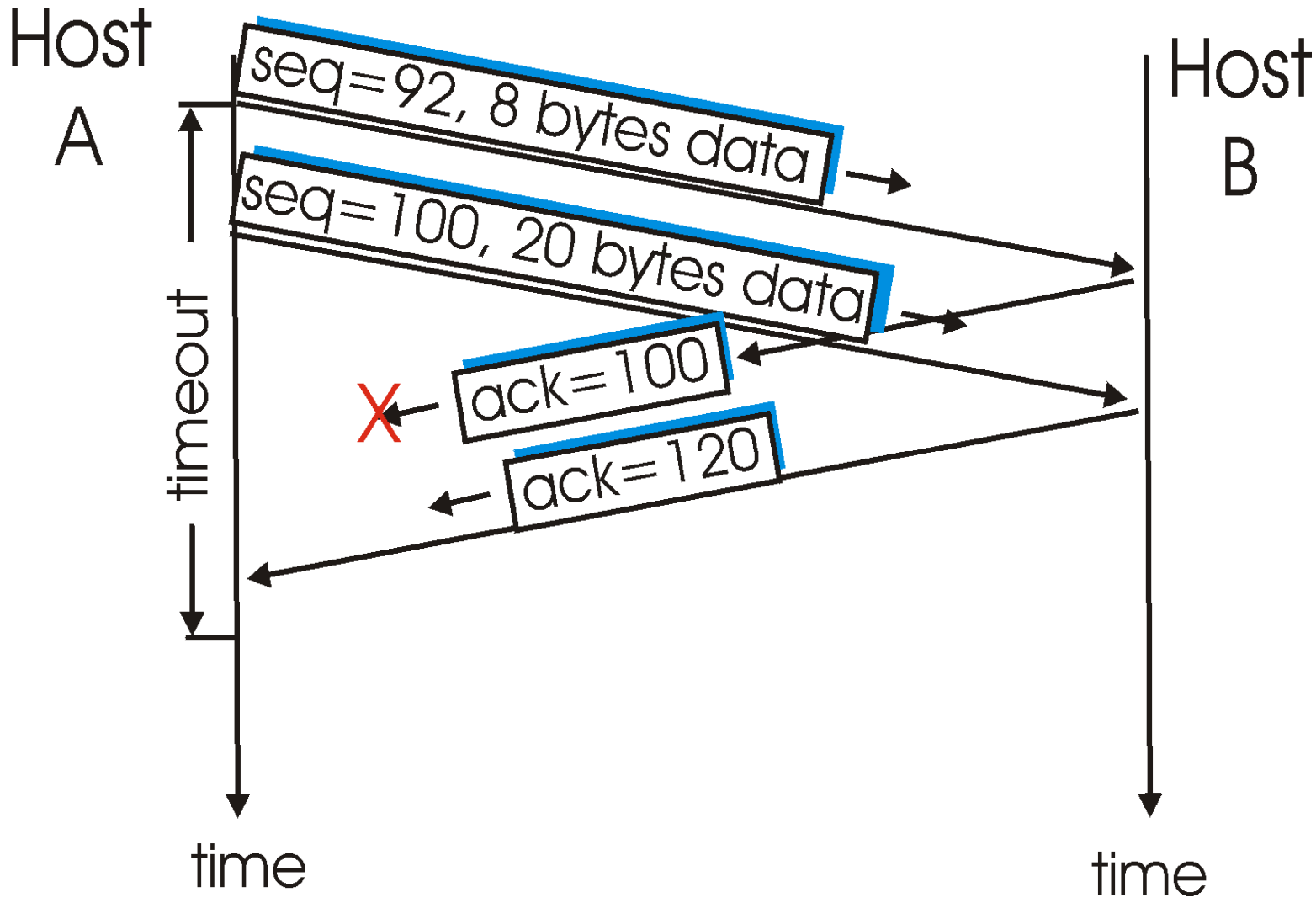
TCP Receiver Actions

Event	Action
Segment arrives with expected SNo, all previous data already ack'ed	Wait up to 500 ms for arrival of another segment. Then send ack
Segment arrives with expected SNo, preceding segment received, but not ack'ed	Send cumulative ack
Out-of-order segment arrives with higher SNo than expected	Send duplicate ack, indicating SNo of next expected byte
Out-of-order segment arrives with lower SNo than expected	Send duplicate ack, indicating SNo of next expected byte

Retransmission Due to Lost Ack



Cumulative Ack Avoids Retransmission



Flow Control

Receiver's buffer has size **`RcvBuffer`**

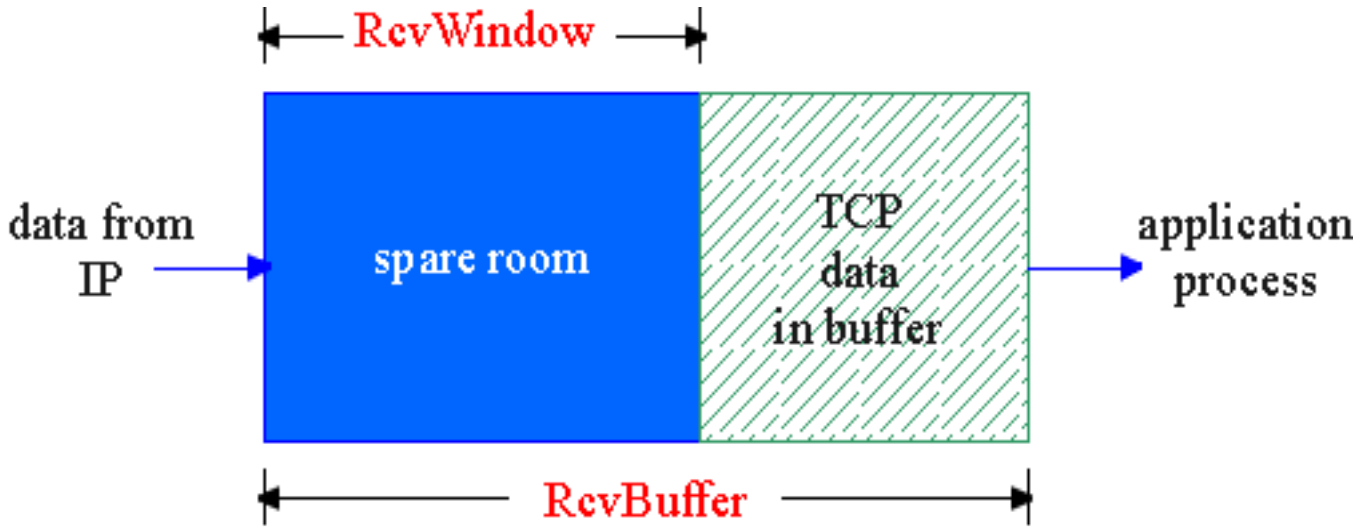
Receiver maintains variables

`LastByteRead`

`LastByteReceived`

Constraint:

`LastByteReceived - LastByteRead <= RcvBuffer`



Flow Control (2)

Receiver communicates to sender

```
RcvWindow =
        RcvBuffer - (LastByteReceived - LastByteRead)
```

Sender maintains variables

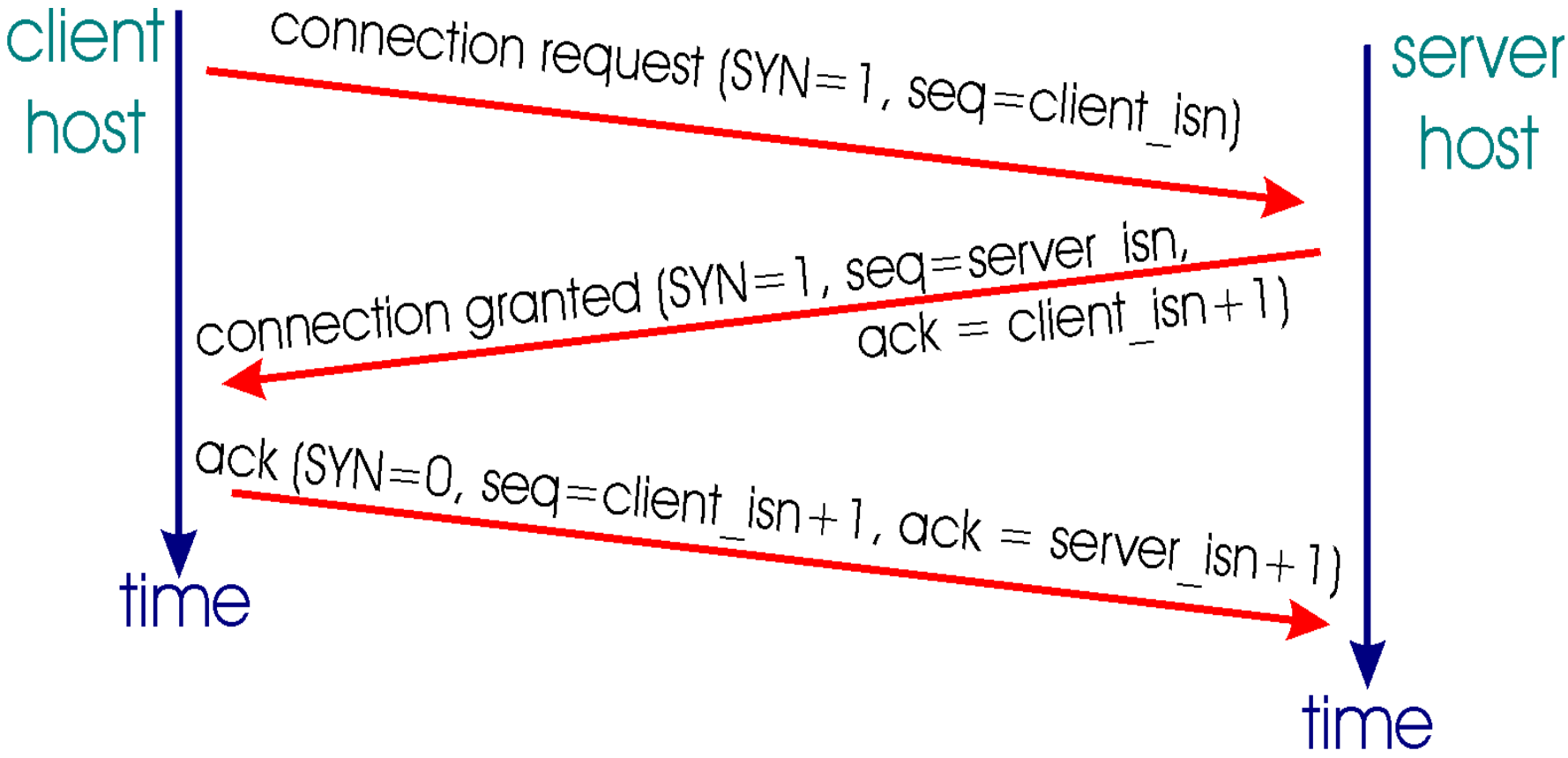
```
LastByteSent
LastByteAcked
```

Sender makes sure

```
LastByteSent - LastByteAcked <= RcvWindow
```

Establishing a TCP Connection

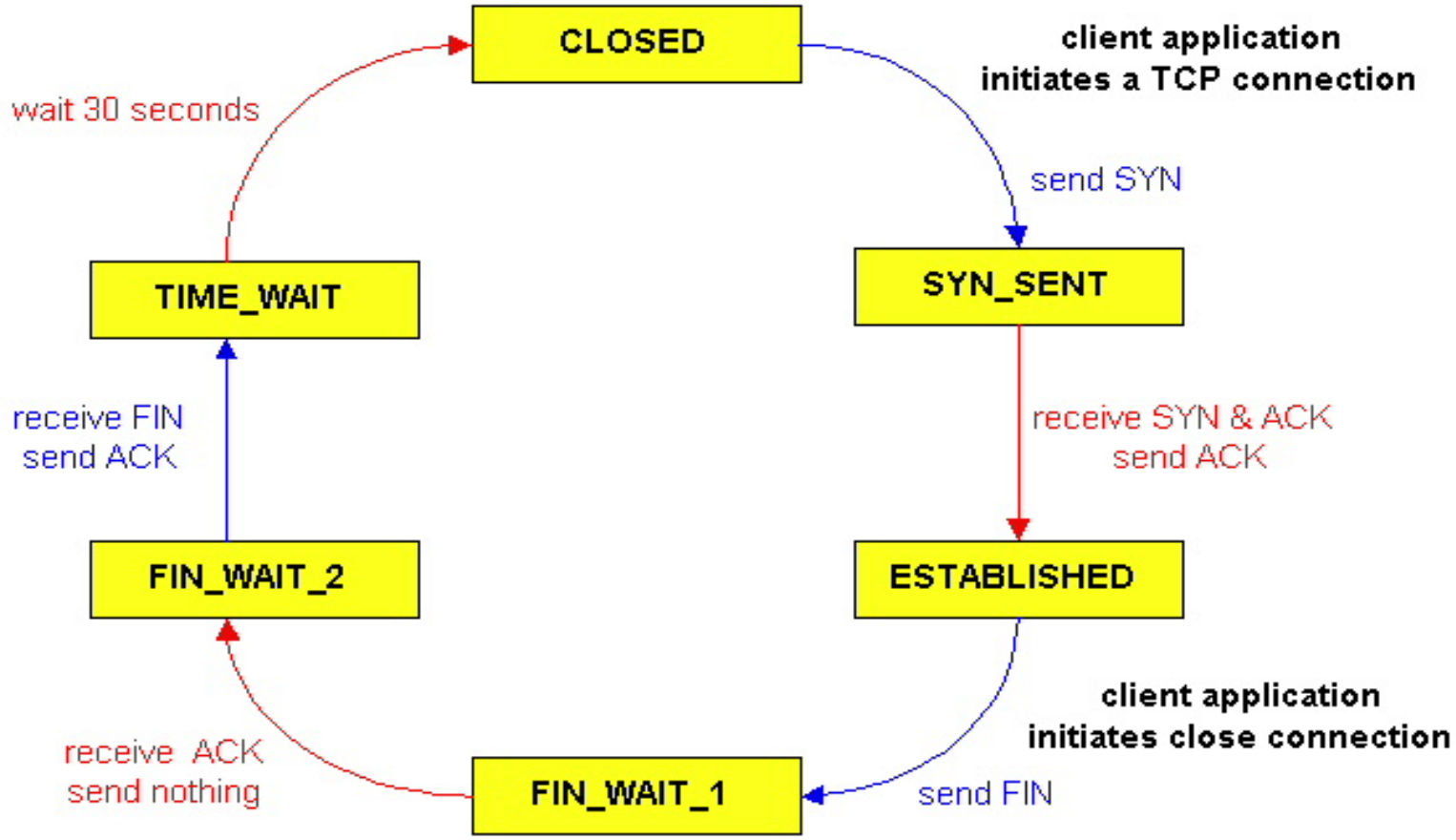
"Three way handshake"



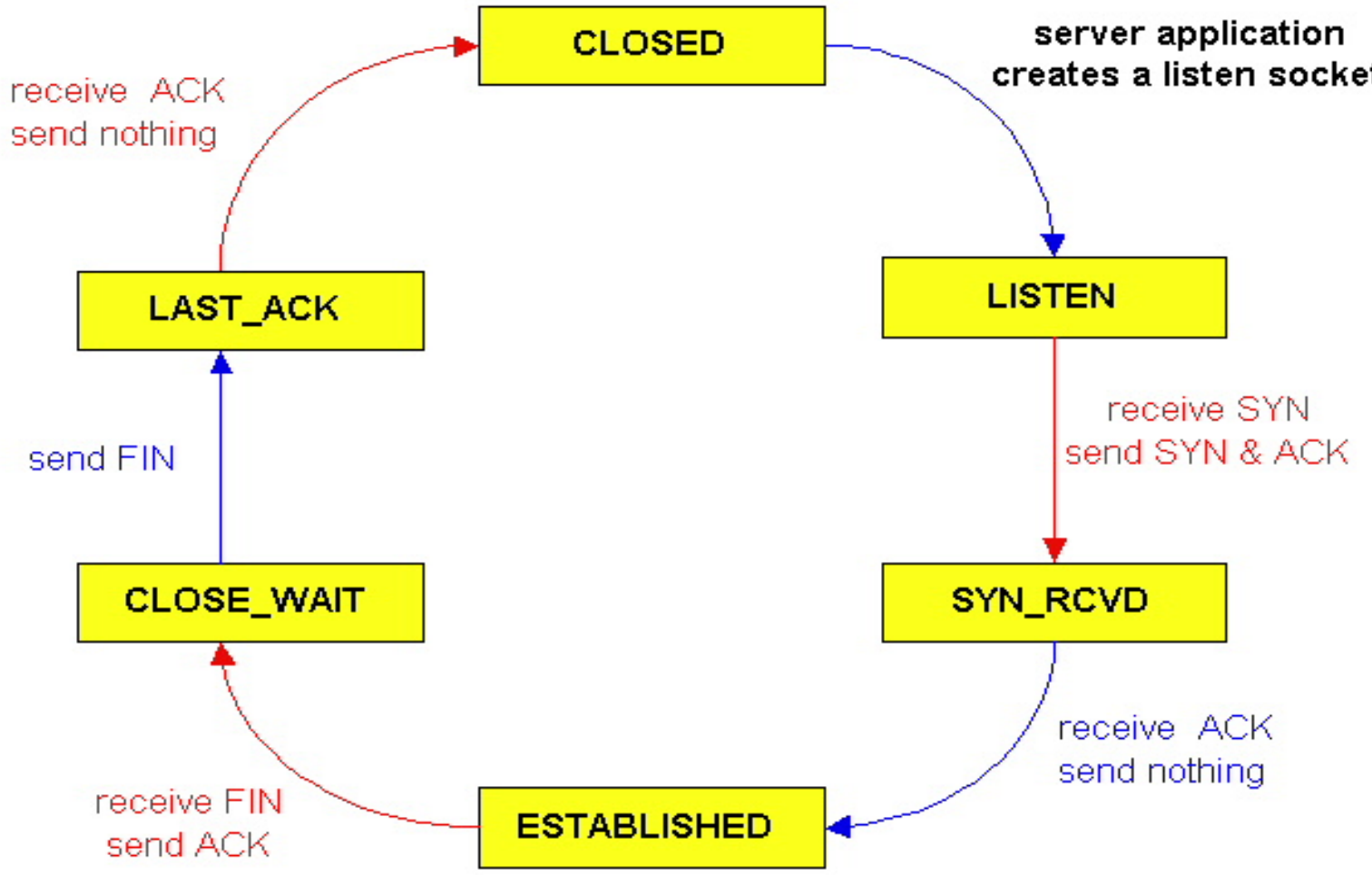
Why are sequence numbers exchanged?

Why does the sender acknowledge?

TCP States of a Client



TCP States of a Server



TCP Summary

- Sequencing
  - messages are divided into segments with sequence numbers
- Flow Control
  - "window" defines number of segments after which receipt needs to be acknowledged
  - acknowlegements
    - piggybacked if there is a reverse data flow,
    - otherwise sent in special segments
- Buffering
  - outgoing buffer with timeout for interactive applications
  - incoming buffer: dropped segments are not acknowledged
- Checksum
- Congestion Control
  - sender starts slowly, exponentially increases number of packets sent

References

In preparing the lectures I have used several sources.

The main ones are the following:

Books:

- Tanenbaum, Computer Networks, 4th Edition, Pearson International
- Coulouris, Dollimore, Kindberg. Distributed Systems – Concepts and Design (CDK)

Slides:

- Andrew Tanenbaum, Slides from his website
- CDK Website
- Marta Kwiatkowska, U Birmingham, slides of course on DS